31 Days Before Your

CCNP and CCIE Enterprise Core Exam

A Day-By-Day Review Guide for the ENCOR 350-401 Certification Exam

ciscopress.com

Patrick Gargano

31 Days Before Your
CCNP and CCIE Enterprise Core Exam

A Day-by-Day Review Guide for the CCNP and CCIE Enterprise Core ENCOR 350-401 Certification Exam

Patrick Gargano

Cisco Press • 221 River Street • Hoboken, NJ 07030 USA

31 Days Before Your CCNP and CCIE Enterprise Core Exam

Patrick Gargano

Copyright © 2021 Pearson Education, Inc.

Published by:
Cisco Press
221 River Street
Hoboken, NJ 07030 USA

All rights reserved. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, request forms, and the appropriate contacts within the Pearson Education Global Rights & Permissions Department, please visit www.pearson.com/permissions.

No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained herein.

ScoutAutomatedPrintCode

Library of Congress Control Number: 2020913346

ISBN-13: 978-0-13-696522-0

ISBN-10: 0-13-696522-9

Warning and Disclaimer

This book is designed to provide information about exam topics for the Cisco Certified Networking Professional (CCNP) Enterprise and Cisco Certified Internetwork Expert (CCIE) Enterprise Infrastructure and Enterprise Wireless certifications. Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied.

The information is provided on an "as is" basis. The author, Cisco Press, and Cisco Systems, Inc., shall have neither liability for nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the discs or programs that may accompany it.

The opinions expressed in this book belong to the author and are not necessarily those of Cisco Systems, Inc.

Trademark Acknowledgments

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Cisco Press or Cisco Systems, Inc., cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Special Sales

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact intlcs@pearson.com.

Feedback Information

At Cisco Press, our goal is to create in-depth technical books of the highest quality and value. Each book is crafted with care and precision, undergoing rigorous development that involves the unique expertise of members from the professional technical community.

Readers' feedback is a natural continuation of this process. If you have any comments regarding how we could improve the quality of this book, or otherwise alter it to better suit your needs, you can contact us through email at feedback@ciscopress.com. Please make sure to include the book title and ISBN in your message.

We greatly appreciate your assistance.

Editor-in-Chief	Mark Taub
Director, ITP Production Management	Brett Bartow
Alliances Manager, Cisco Press	Arezou Gol
Executive Editor	James Manly
Managing Editor	Sandra Schroeder
Development Editor	Ellie Bru
Project Editor	Mandie Frank
Copy Editor	Kitty Wilson
Technical Editor	Akhil Behl
Editorial Assistant	Cindy Teeters
Designer	Chuti Prasertsith
Composition	codeMantra
Indexer	Ken Johnson
Proofreader	Donna E. Mulder

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV Amsterdam,
The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at **www.cisco.com/go/offices.**

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: www.cisco.com/go/trademarks. Third party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

Credits

Figure Number	Credit/Attribution
Figure 3-2	Screenshot of Ansible Playbook Example © 2020 Red Hat, Inc.
Figure 3-3	Screenshot of Ansible Inventory Example © 2020 Red Hat, Inc.
Figure 3-4	Screenshot of Ansible Playbook Results © 2020 Red Hat, Inc.

NIST SP 800-82 Rev. 2, Guide to Industrial Control Systems (ICS) Security, August 12, 2015

About the Author

Patrick Gargano has been an educator since 1996, a Cisco Networking Academy Instructor since 2000, and a Certified Cisco Systems Instructor (CCSI) since 2005. He is currently working for Cisco as a Content Engineer on the Enterprise Technical Education team within DevCX. Until recently, he was based in Australia, where he worked as a Content Development Engineer at Skyline ATS, responsible for CCNP Enterprise course development with Learning@Cisco. He previously led the Networking Academy program at Collège La Cité in Ottawa, Canada, where he taught CCNA/CCNP-level courses, and he has also worked for Cisco Learning Partners NterOne and Fast Lane UK. In 2018 Patrick was awarded the Networking Academy Above and Beyond Instructor award for leading CCNA CyberOps early adoption and instructor training in Quebec, Canada. Patrick has also twice led the Cisco Networking Academy Dream Team at Cisco Live US. His previous Cisco Press publications include *CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide* (2020), *31 Days Before Your CCNA Security Exam* (2016), and *CCNP Routing and Switching Portable Command Guide* (2014). His certifications include CCNA, CyberOps Associate, and CCNP Enterprise, as well as the Enterprise Core and Enterprise Advanced Infrastructure Implementation specialists. He holds BEd and BA degrees from the University of Ottawa, and he is completing a master of professional studies (MPS) degree in computer networking at Fort Hays State University.

About the Technical Reviewer

Akhil Behl is a Pre-Sales Manager with a leading service provider. His technology portfolio encompasses IoT, collaboration, security, infrastructure, service management, cloud, and data center. He has over 12 years of experience working in leadership, advisory, business development, and consulting positions with various organizations and leading global accounts, driving toward business innovation and excellence. Previously, he was in a leadership role with Cisco Systems.

Akhil has a bachelor of technology degree in electronics and telecommunications from IP University, India, and a master's degree in business administration from Symbiosis Institute, India. Akhil holds dual CCIE certifications in Collaboration and Security, PMP, ITIL, VCP, TOGAF, CEH, ISO/IEC 27002, and many other industry certifications.

He has published several research papers in national and international journals, including IEEE publications, and has been a speaker at prominent industry forums such as Interop, Enterprise Connect, Cloud Connect, Cloud Summit, Cisco Sec-Con, IT Expo, Computer Society of India, Singapore Computer Society, and Cisco Networkers.

Akhil is the author of several Cisco Press books. He also is a technical editor for Cisco Press and other publications. Akhil can be reached at akbehl@technologist.com.

Dedications

This book is dedicated to my wife, Kathryn, and our son, Samuel, without whose love and support none of this would be possible. To my mother and Albert, thank you for always encouraging and believing in me. To my sister, Martina, your new Zen life is something to behold and an inspiration to me.

Acknowledgments

When James Manly and Allan Johnson reached out to me in March 2020 to see if I would be interested in authoring a *31 Days* book for the ENCOR 350-401 certification exam, I was initially a bit daunted because of the breadth and depth of the blueprint, but after spending the previous year working with Learning@Cisco to develop material for ENARSI, ENCOR, and ENSLD courses, I felt I could take up the challenge.

As the 2020 global pandemic hit, writing a book from home turned out to be the ideal project for me and my family, so my first thanks must go to both James and Allan for trusting me with this contribution to the *31 Days* series.

My technical editor, Akhil Behl, kept me on my toes and was quick with his comments, ensuring that we could get this book out to you quickly.

My development editor, Ellie Bru, did a fabulous job of cajoling and encouraging me throughout the writing process. She is a gem; Cisco Press should count themselves lucky to have her. My thanks also go out to Mandie Frank and Kitty Wilson, who ensured that the book you now hold in your hands looks good and reads easily.

I think I've developed a reputation at Cisco Press for being a bit difficult when it comes to choosing the photo for the cover. Thank you to Chuti Prasertsith for his patience as we came up with the final design you now see.

Contents at a Glance

Introduction xxx

Day 31: Enterprise Network Architecture 1

Day 30: Packet Switching and Forwarding 15

Day 29: LAN Connectivity 33

Day 28: Spanning Tree Protocol 55

Day 27: Port Aggregation 91

Day 26: EIGRP 105

Day 25: OSPFv2 121

Day 24: Advanced OSPFv2 and OSPFv3 143

Day 23: BGP 163

Day 22: First-Hop Redundancy Protocols 183

Day 21: Network Services 205

Day 20: GRE and IPsec 227

Day 19: LISP and VXLAN 251

Day 18: SD-Access 269

Day 17: SD-WAN 285

Day 16: Multicast 303

Day 15: QoS 323

Day 14: Network Assurance, Part 1 349

Day 13: Network Assurance, Part 2 373

Day 12: Wireless Concepts 389

Day 11: Wireless Deployment 407

Day 10: Wireless Client Roaming and Authentication 433

Day 9: Secure Network Access 465

Day 8: Infrastructure Security 491

Day 7: Virtualization 517

Day 6: Cisco DNA Center 533

Day 5: Network Programmability 555

Day 4: REST APIs 583

Day 3: Network Automation 599

Day 2: Review Lab 1 621

Day 1: Review Lab 2 633

Index 639

Contents

Introduction xxx

Day 31: Enterprise Network Architecture 1

ENCOR 350-401 Exam Topics 1

Key Topics 1

Hierarchical LAN Design Model 1
  Access Layer 2
  Distribution Layer 3
  Core Layer 4

Enterprise Network Architecture Options 4
  Two-Tier Design (Collapsed Core) 4
  Three-Tier Design 5
  Layer 2 Access Layer (STP Based): Loop-Free and Looped 6
  Layer 3 Access Layer (Routed Based) 7
  Simplified Campus Design Using VSS and StackWise 9
  Common Access–Distribution Interconnection Designs 10
  Software-Defined Access (SD-Access) Design 11
  Spine-and-Leaf Architecture 12

Study Resources 13

Day 30: Packet Switching and Forwarding 15

ENCOR 350-401 Exam Topics 15

Key Topics 15

Layer 2 Switch Operation 15
  MAC Address Table and TCAM 17

Layer 3 Switch Operation 17

Forwarding Mechanisms 19
  Control and Data Plane 19
  Cisco Switching Mechanisms 20
  Process and Fast Switching 22
  Cisco Express Forwarding 24
    Centralized and Distributed Switching 26
  Hardware Redundancy Mechanisms 28
    Cisco Nonstop Forwarding 29
  SDM Templates 30

Study Resources 32

Day 29: LAN Connectivity 33

- ENCOR 350-401 Exam Topics 33
- Key Topics 33
- VLAN Overview 33
  - Creating a VLAN 35
- Access Ports 35
- 802.1Q Trunk Ports 38
  - Native VLAN 40
  - Allowed VLANs 40
  - 802.1Q Trunk Configuration 41
  - 802.1Q Trunk Verification 41
- Dynamic Trunking Protocol 44
  - DTP Configuration Example 45
- VLAN Trunking Protocol 46
  - VTP Modes 47
  - VTP Configuration Revision 48
  - VTP Versions 48
  - VTP Configuration Example 49
- Inter-VLAN Routing 50
  - Inter-VLAN Routing Using an External Router 51
  - Inter-VLAN Routing Using Switched Virtual Interfaces 52
  - Routed Switch Ports 53
- Study Resources 54

Day 28: Spanning Tree Protocol 55

- ENCOR 350-401 Exam Topics 55
- Key Topics 55
- IEEE 802.1D STP Overview 55
  - STP Operations 57
  - Bridge Protocol Data Unit 57
  - Root Bridge Election 59
  - Root Port Election 60
  - Designated Port Election 62
  - STP Port States 63
- Rapid Spanning Tree Protocol 64
  - RSTP Port Roles 64
  - RSTP Port States 65

    - RSTP Rapid Transition to Forwarding State 66
        - Edge Ports 66
        - Link Type 66
    - RSTP Synchronization 66
    - RSTP Topology Change 67
- STP and RSTP Configuration and Verification 68
    - Changing STP Bridge Priority 71
    - STP Path Manipulation 74
    - Enabling and Verifying RSTP 76
- STP Stability Mechanisms 77
    - STP PortFast and BPDU Guard 77
    - Root Guard 79
    - STP Loop Guard 80
    - Unidirectional Link Detection 82
- Multiple Spanning Tree Protocol 83
    - MST Regions 84
    - MST Instances 85
    - MST Configuration and Verification 87
    - Configuring MST Path Cost and Port Priority 89
- Study Resources 90

Day 27: Port Aggregation 91

- ENCOR 350-401 Exam Topics 91
- Key Topics 91
- Need for EtherChannel 91
- EtherChannel Mode Interactions 93
    - LACP 94
    - PAgP 94
    - Static 95
- EtherChannel Configuration Guidelines 95
- EtherChannel Load Balancing Options 96
- EtherChannel Configuration and Verification 97
- Advanced EtherChannel Tuning 101
    - LACP Hot-Standby Ports 101
    - Configuring the LACP Max Bundle Feature 102
    - Configuring the LACP Port Channel Min-Links Feature 103
    - Configuring the LACP System Priority 103

Configuring the LACP Port Priority 103
Configuring LACP Fast Rate Timer 103

Study Resources 104

Day 26: EIGRP 105

ENCOR 350-401 Exam Topics 105

Key Topics 105

EIGRP Features 105

EIGRP Reliable Transport Protocol 106
EIGRP Operation Overview 107
EIGRP Packet Format 108

Establishing EIGRP Neighbor Adjacency 109

EIGRP Metrics 110
EIGRP Wide Metrics 112

EIGRP Path Selection 115
Loop-Free Path Selection 116

EIGRP Load Balancing and Sharing 117
Equal-Cost Load Balancing 118
Unequal-Cost Load Balancing 118

Study Resources 119

Day 25: OSPFv2 121

ENCOR 350-401 Exam Topics 121

Key Topics 121

OSPF Characteristics 121

OSPF Process 123

OSPF Neighbor Adjacencies 125

Building a Link-State Database 127

OSPF Neighbor States 127

OSPF Packet Types 129

OSPF LSA Types 130

Single-Area and Multiarea OSPF 131

OSPF Area Structure 132

OSPF Network Types 133

- OSPF DR and BDR Election 134
- OSPF Timers 136
- Multiarea OSPF Configuration 137
- Verifying OSPF Functionality 138
- Study Resources 141

Day 24: Advanced OSPFv2 and OSPFv3 143

- ENCOR 350-401 Exam Topics 143
- Key Topics 143
- OSPF Cost 143
  - Shortest Path First Algorithm 144
- OSPF Passive Interfaces 145
- OSPF Default Routing 146
- OSPF Route Summarization 147
  - OSPF ABR Route Summarization 148
  - Summarization on an ASBR 149
  - OSPF Summarization Example 149
- OSPF Route Filtering Tools 151
  - Distribute Lists 151
  - OSPF Filtering Options 152
    - OSPF Filtering: Filter List 152
    - OSPF Filtering: Area Range 153
    - OSPF Filtering: Distribute List 154
    - OSPF Filtering: Summary Address 154
- OSPFv3 155
  - OSPFv3 LSAs 156
- OSPFv3 Configuration 157
  - OSPFv3 Verification 161
- Study Resources 162

Day 23: BGP 163

- ENCOR 350-401 Exam Topics 163
- Key Topics 163
- BGP Interdomain Routing 163
  - BGP Characteristics 164
  - BGP Path Vector Functionality 165
  - BGP Routing Policies 166
- BGP Multihoming 167

- BGP Operations 169
  - BGP Data Structures 169
  - BGP Message Types 169
- BGP Neighbor States 171
- BGP Neighbor Relationships 172
  - EBGP and IBGP 172
- BGP Path Selection 173
  - BGP Route Selection Process 174
- BGP Path Attributes 175
  - Well-Known BGP Attributes 175
    - Well-Known Mandatory Attributes 175
    - Well-Known Discretionary Attributes 175
  - Optional BGP Attributes 176
    - Optional Transitive Attributes 176
    - Optional Nontransitive Attributes 176
- BGP Configuration 176
  - Verifying EBGP 178
- Study Resources 182

Day 22: First-Hop Redundancy Protocols 183

- ENCOR 350-401 Exam Topics 183
- Key Topics 183
- Default Gateway Redundancy 183
- First Hop Redundancy Protocol 184
- HSRP 186
  - HSRP Group 187
  - HSRP Priority and HSRP Preempt 189
  - HSRP Timers 190
  - HSRP State Transition 191
  - HSRP Advanced Features 191
    - HSRP Object Tracking 192
    - HSRP Multigroup 194
    - HSRP Authentication 195
  - HSRP Versions 196
  - HSRP Configuration Example 197
- VRRP 199
  - VRRP Authentication 201
  - VRRP Configuration Example 201
- Study Resources 203

Day 21: Network Services 205

ENCOR 350-401 Exam Topics 205

Key Topics 205

Network Address Translation 205
  NAT Address Types 207
  NAT Implementation Options 208
    Static NAT 209
    Dynamic NAT 210
    Port Address Translation (PAT) 211
  NAT Virtual Interface 213
  NAT Configuration Example 214
  Tuning NAT 217

Network Time Protocol 218
  NTP Versions 218
  NTP Modes 219
    NTP Server 220
    NTP Client 220
    NTP Peer 221
    Broadcast/Multicast 221
  NTP Source Address 222
  Securing NTP 222
    NTP Authentication 222
    NTP Access Lists 222
  NTP Configuration Example 223

Study Resources 226

Day 20: GRE and IPsec 227

ENCOR 350-401 Exam Topics 227

Key Topics 227

Generic Routing Encapsulation 227
  GRE Configuration Steps 228
  GRE Configuration Example 229

IP Security (IPsec) 232
  Site-to-Site VPN Technologies 232
    Dynamic Multipoint VPN 234
    Cisco IOS FlexVPN 235
  IPsec VPN Overview 235
  IP Security Services 236
  IPsec Security Associations 239

    - IPsec: IKE 240
        - IKEv1 Phase 1 240
        - IKEv1 Phase 2 241
        - IKEv2 242
    - IPsec Site-to-Site VPN Configuration 242
        - GRE over IPsec Site-to-Site VPNs 243
        - Site-to-Site Virtual Tunnel Interface over IPsec 247
- Study Resources 250

Day 19: LISP and VXLAN 251

- ENCOR 350-401 Exam Topics 251
- Key Topics 251
- Locator/ID Separation Protocol 251
    - LISP Terms and Components 253
    - LISP Data Plane 255
    - LISP Control Plane 258
    - LISP Host Mobility 259
    - LISP Host Mobility Deployment Models 259
        - LISP Host Mobility with an Extended Subnet 259
        - LISP Host Mobility Across Subnets 260
        - LISP Host Mobility Example 260
- Virtual Extensible LAN (VXLAN) 262
    - VXLAN Encapsulation 263
    - VXLAN Gateways 265
    - VXLAN-GPO Header 266
- Study Resources 268

Day 18: SD-Access 269

- ENCOR 350-401 Exam Topics 269
- Key Topics 269
- Software-Defined Access 269
    - Need for Cisco SD-Access 270
    - Cisco SD-Access Overview 272
        - Cisco SD-Access Fabric 272
        - Fabric Overlay Types 274
        - Fabric Underlay Provisioning 274
    - Cisco SD-Access Fabric Data Plane and Control Plane 275
    - Cisco SD-Access Fabric Policy Plane 275
        - Cisco TrustSec and ISE 276

- Cisco SD-Access Fabric Components 277
  - Cisco SD-Access Control Plane Node 277
  - Cisco SD-Access Edge Node 278
  - Cisco SD-Access Border Node 278
  - Cisco SD-Access Intermediate Node 279
  - Cisco SD-Access Wireless LAN Controller and Fabric Mode Access Points (APs) 279
- Shared Services in Cisco SD-Access 281
  - Fusion Router 282

Study Resources 283

Day 17: SD-WAN 285

ENCOR 350-401 Exam Topics 285

Key Topics 285

Software-Defined WAN 285
- Need for Cisco SD-WAN 286
- SD-WAN Architecture and Components 287
  - SD-WAN Orchestration Plane 288
  - SD-WAN Management Plane 289
  - SD-WAN Control Plane 290
  - SD-WAN Data Plane 291
  - SD-WAN Automation and Analytics 293
- Cisco SD-WAN Application Performance Optimization 294

Cisco SD-WAN Solution Example 295
- Site ID 296
- System IP 296
- Organization Name 296
- Public and Private IP Addresses 297
- TLOC 297
- Color 297
- Overlay Management Protocol (OMP) 298
- Virtual Private Networks (VPNs) 298

Cisco SD-WAN Routing 299

Study Resources 301

Day 16: Multicast 303

ENCOR 350-401 Exam Topics 303

Key Topics 303

Multicast Overview 303
- Unicast vs. Multicast 304

    - Multicast Operations 305
    - Multicast Benefits and Drawbacks 306
    - IP Multicast Applications 307
    - IP Multicast Group Address 307
    - IP Multicast Service Model 310
- Internet Group Management Protocol 311
    - IGMPv1 Overview 311
    - IGMPv2 Overview 312
    - IGMPv3 Overview 312
- Multicast Distribution Trees 313
    - Source Trees 313
    - Shared Trees 314
    - Source Trees vs. Shared Trees 315
- IP Multicast Routing 315
    - Protocol Independent Multicast 316
        - PIM-DM Overview 316
        - PIM-SM Overview 317
    - Rendezvous Point 319
        - Static RP 319
        - PIM Bootstrap Router 320
        - Auto-RP 320
- Study Resources 321

Day 15: QoS 323

- ENCOR 350-401 Exam Topics 323
- Key Topics 323
- Quality of Service 323
    - Need for Quality of Service 323
    - Converged Networks 324
        - Components of Network Delay 325
        - Jitter 326
        - Dejitter Buffer Operation 326
        - Packet Loss 327
- QoS Models 327
    - Best-Effort QoS Model 328
    - IntServ Model 328
    - DiffServ Model 328
- QoS Mechanisms Overview 329
    - Classification and Marking 330
        - Classification 330
        - Marking 332

Layer 2 Classification and Marking 334
802.1p Class of Service 334
802.11 Wireless QoS: 802.11e 336
Layer 3 Marking: IP Type of Service 337
Layer 3 Marking: DSCP Per-Hop Behaviors 338
Mapping Layer 2 to Layer 3 Markings 339
Mapping Markings for Wireless Networks 340
Policing, Shaping, and Re-marking 340
Managing Congestion 342
Class-Based Weighted Fair Queuing 343
Tools for Congestion Avoidance 344
QoS Policy 345
Define an Overall QoS Policy 345
Methods for Implementing a QoS Policy 346

Study Resources 348

Day 14: Network Assurance, Part 1 349

ENCOR 350-401 Exam Topics 349

Key Topics 349

Troubleshooting Concepts 349
Diagnostic Principles 350
Network Troubleshooting Procedures: Overview 351

Network Diagnostic Tools 352
Using the ping Command 352
The Extended Ping 353
Using traceroute 355
Using Debug 357
The Conditional Debug 358

Cisco IOS IP SLAs 360
IP SLA Source and Responder 362

Switched Port Analyzer Overview 365
Local SPAN 366
Local SPAN Configuration 366
Verify the Local SPAN Configuration 367
Remote SPAN 368
RSPAN Configuration 369
Verify the Remote SPAN Configuration 369
Encapsulated Remote SPAN 370
ERSPAN Configuration 371
ERSPAN Verification 372

Study Resources 372

Day 13: Network Assurance, Part 2 373

- ENCOR 350-401 Exam Topics 373
- Key Topics 373
- Logging Services 373
- Understanding Syslog 374
    - Syslog Message Format and Severity 375
- Simple Network Management Protocol 376
    - SNMP Operations 377
- NetFlow 378
    - Creating a Flow in the NetFlow Cache 379
    - NetFlow Data Analysis 380
    - NetFlow Export Data Format 381
    - Traditional NetFlow Configuration and Verification 382
    - Flexible NetFlow 384
        - Traditional vs. Flexible NetFlow 384
        - Flexible NetFlow Configuration and Verification 385
- Study Resources 387

Day 12: Wireless Concepts 389

- ENCOR 350-401 Exam Topics 389
- Key Topics 389
- Explain RF Principles 389
    - RF Spectrum 389
    - Frequency 390
    - Wavelength 391
    - Amplitude 391
    - Free Path Loss 392
    - RSSI and SNR 393
        - RSSI 393
        - SNR 394
    - Watts and Decibels 395
    - Antenna Power 397
        - Effective Isotropic-Radiated Power 398
- IEEE Wireless Standards 399
    - 802.11 Standards for Channels and Data Rates 399
        - 802.11b/g 399
        - 802.11a 400
        - 802.11n 400
        - 802.11ac 400
        - 802.11ax (Wi-Fi 6) 401

        - 802.11n/802.11ac MIMO 401
            - Maximal Ratio Combining 402
            - Beamforming 403
            - Spatial Multiplexing 404
        - 802.11ac MU-MIMO 405
    - Study Resources 406

Day 11: Wireless Deployment 407

- ENCOR 350-401 Exam Topics 407
- Key Topics 407
- Wireless Deployment Overview 407
    - Autonomous AP Deployment 408
        - Autonomous Deployment Traffic Flow 409
    - Centralized Cisco WLC Deployment 410
        - Split MAC 411
        - CAPWAP 413
        - Centralized Deployment Traffic Flow 413
    - FlexConnect Deployment 415
    - Cloud-Managed Meraki Deployment 416
        - Cisco Meraki Deployment Traffic Flow 417
    - Cisco Catalyst 9800 Series Controller Deployment Options 417
        - Catalyst 9800 Wireless Controller for Cloud 419
        - Catalyst 9800 Embedded Wireless Controller 419
    - Cisco Mobility Express 422
- Wireless AP Operation 422
    - Wireless LAN Controller Discovery Process 422
        - AP Join Order 423
        - AP Failover 423
    - AP Modes 425
        - Local Mode 425
        - FlexConnect Mode 425
        - Bridge Mode 427
        - Other Modes 427
- Antenna Characteristics 428
    - Antenna Types 428
        - Omnidirectional Antennas 429
        - Directional Antennas 430
- Study Resources 431

Day 10: Wireless Client Roaming and Authentication 433

ENCOR 350-401 Exam Topics 433

Key Topics 433

Wireless Roaming 434
  Mobility Groups and Domains 435
  Wireless Roaming Types 436
    Layer 2 Roaming: Centralized Controllers 437
    Layer 3 Roaming: Centralized Controllers 439
  Layer 3 Inter-Controller Roaming Example 439
  Roaming with Auto-Anchor Mobility (Guest Access) 441

Wireless Location Services 442
  Cisco CMX 442
  Cisco CMX Analytics Tools 444
    Presence Analytics 444
    Location Analytics 444
  Location Accuracy 444

Wireless Client Authentication 445
  Pre-Shared Key Authentication 446
    PSK Authentication Process 447
    WPA2 and WPA3 PSK Authentication Example 447
  802.1X User Authentication Overview 449
    Extensible Authentication Protocol 452
    802.1X EAP Authentication Example 453
  Guest Access with Web Auth 457
    Local Web Authentication 458
    Local Web Authentication with Auto-Anchor 458
    Local Web Portal with External Authentication 459
    Centralized Web Authentication 459
    Web Auth Authentication Configuration Example 459

Study Resources 463

Day 9: Secure Network Access 465

ENCOR 350-401 Exam Topics 465

Key Topics 465

Network Security Threatscape 465

Network Security Components 468
  Intrusion Prevention Systems 468
  Virtual Private Networks 469
  Content Security 470

Endpoint Security 471
    Centralized Endpoint Policy Enforcement 472
    Cisco AMP for Endpoints 473
Firewall Concepts 476
    Next-Generation Firewalls 478
TrustSec 478
    Inline SGT Transport 480
    Security Group Firewall 482
MACsec 483
Identity Management 484
    802.1X for Wired and Wireless Endpoint Authentication 485
    MAC Authentication Bypass 487
    Web Authentication 489

Study Resources 490

Day 8: Infrastructure Security 491

ENCOR 350-401 Exam Topics 491

Key Topics 491

Secure Access Control 491
    Securing Device Access 492
        Enable Secret Password 492
        Line Password 493
        Username and Password 494
    AAA Framework Overview 495
        RADIUS and TACACS+ 496
    Authentication Options 498
    Enabling AAA and Configuring a Local User 499
    Configuring RADIUS for Console and vty Access 500
    Configuring TACACS+ for Console and vty Access 501
    Configuring Authorization and Accounting 501

Access Control Lists 504
    ACL Overview 504
    ACL Wildcard Masking 505
        Wildcard Bit Mask Abbreviations 507
    Types of ACLs 507
        Configuring Numbered Access Lists 508
        Configuring Numbered Extended IPv4 ACLs 509
        Configuring Named Standard ACLs 510
        Configuring Named Extended ACLs 511
    Applying ACLs to Interfaces 511

Control Plane Policing 513
    CoPP Configuration 514
Study Resources 516

Day 7: Virtualization 517

ENCOR 350-401 Exam Topics 517
Key Topics 517
Server Virtualization 517
    Physical Server 518
    Virtualized Server 518
    Basic Virtualized Server Environment 519
    Hypervisor: Abstraction Layer 520
        Type 1 Hypervisors 521
        Type 2 Hypervisors 521
    VM Definition 522
    Managing Virtual Machines 522
Network Function Virtualization 523
    Cisco Enterprise NFV Solution Architecture 523
        NFVIS Building Blocks 524
    Cisco NFV Hardware Options 526
Network Path Isolation 527
    Layer 2 and Layer 3 Virtualization 527
    Virtual Routing and Forwarding 529
    Configuring and Verifying VRF-Lite 530
Study Resources 532

Day 6: Cisco DNA Center 533

ENCOR 350-401 Exam Topics 533
Key Topics 533
Need for Digital Transformation 533
Cisco Digital Network Architecture 535
Cisco Intent-Based Networking 537
    IBN Building Blocks 537
Cisco DNA Center Features 539
    Cisco DNA Center Assurance 541
    Cisco DNA Center Automation Workflow 542
        Network Discovery and Management 542
        Inventory Management 543
        Software Image Management 544

        IP Address Pools 545
        Network Hierarchy 546
        Day 0 Network Provisioning 546
        Day *N* Network Automation 547
    Cisco DNA Assurance Workflow 548
        Cisco DNA Center Assurance: AI-Driven Data 550
        Cisco DNA Center Assurance: Client 360 551
        Cisco DNA Center Assurance: Application Health 552
Study Resources 554

Day 5: Network Programmability 555

ENCOR 350-401 Exam Topics 555

Key Topics 555

Python Concepts 555
    Execute Python Code 556
        Using the Dynamic Interpreter 556
        Writing Python Scripts 557
    Python Helper Utilities and Functions 557
    Writing Idiomatic Python 558
    Common Python Data Types 559
        String Data Types 559
        Numbers Data Types 561
        Boolean Data Types 562
    Describe Conditionals 562
    Script Writing and Execution 564
        Shebang 564
        Script Entry Point 564

Device Management and Network Programmability 565
    Data Encoding Formats 567
        JSON 567
        XML 570
    Data Models 572
        YANG 573
        REST 574
        NETCONF 575

Study Resources 582

Day 4: REST APIs 583

ENCOR 350-401 Exam Topics 583

Key Topics 583

Application Programming Interfaces 583

Southbound APIs 584
Northbound APIs 584
REST API Response Codes and Results 585
HTTP Status Codes 586
REST API Security 586
REST APIs in Cisco DNA Center 587
Intent API 588
Know Your Network 588
Site Management 588
Operational Tools 589
Authentication 589
Multivendor Support 589
Integration API 590
REST APIs in Cisco vManage 590
Resource Data 591
Cisco SD-WAN API Library and Documentation 592
Performing REST API Operations on a vManage Web Server Using Python 594
Performing REST API Operations on a vManage Web Server Using Postman 596

Study Resources 598

Day 3: Network Automation 599

ENCOR 350-401 Exam Topics 599

Key Topics 599

Configuration Management Tools 599
Configuration Management for Networking 600
System Management with Ansible 602
System Management with Ansible: Components 602
System Management with Ansible: Tools 603
How Ansible Works 603
How Ansible Works: Push Model 603
Ansible Playbooks: Terms 604
Ansible Playbooks: Components 604
Ansible Playbooks: Inventory File 605
Ansible: Executing the Playbooks 606
System Management with Puppet 607
Puppet Architecture 607
Basic Puppet Concepts 607
Puppet Example 608
System Management with Chef 610
Chef Concepts 610
Chef Example 611

        System Management with SaltStack 612
            Salt Architecture 612
            SaltStack Example 614
    Embedded Events Manager 615
        EEM Architecture 615
            Policies 616
            EEM Server 616
            Event Detectors 616
        Writing EEM Policies 617
            EEM Applet 617
            EEM Script 618
        Writing an EEM Policy Using the Cisco IOS CLI 618
        Using EEM and Tcl Scripts 619
    Study Resources 620

Day 2: Review Lab 1 621

    Objective 621
        Topology 621
        Addressing Table 623
        Tasks 623
        Part 1: Build the Network and Configure Basic Device Settings and Interface Addressing 624
        Part 2: Configure the Layer 2 Network and Host Support 624
        Part 3: Configure Routing Protocols 625
        Part 4: Configure First-Hop Redundancy and IP SLA Functionality 627
        Part 5: Configure Secure Access 630
        Part 6: Configure Network Management Features 631

Day 1: Review Lab 2 633

    Objective 633
        Topology 633
        Addressing Table 634
        Tasks 635
        Part 1: Build the Network and Configure Basic Device Settings 635
        Part 2: Configure VRF and Static Routing 636
        Part 3: Configure Layer 2 Network 637
        Part 4: Configure Secure Access 637

Index 639

Command Syntax Conventions

The conventions used to present command syntax in this book are the same conventions used in the IOS Command Reference. The Command Reference describes these conventions as follows:

- **Boldface** indicates commands and keywords that are entered literally as shown. In actual configuration examples and output (not general command syntax), boldface indicates commands that are manually input by the user (such as a **show** command).
- *Italic* indicates arguments for which you supply actual values.
- Vertical bars (|) separate alternative, mutually exclusive elements.
- Square brackets ([ ]) indicate an optional element.
- Braces ({ }) indicate a required choice.
- Braces within brackets ([{ }]) indicate a required choice within an optional element.

Reader Services

Register your copy of this book at www.ciscopress.com/title/9780136965220 for convenient access to downloads, updates, and corrections as they become available. To start the registration process, go to www.ciscopress.com/register and log in or create an account. (Be sure to check the box indicating that you would like to hear from us to receive exclusive discounts on future editions of this product.) Enter the product ISBN **9780136965220** and click Submit. When the process is complete, you can find any available bonus content under Registered Products.

Introduction

If you're reading this Introduction, you've probably already spent a considerable amount of time and energy pursuing the ENCOR 350-401 exam. Regardless of how you got to this point in your CCNP/CCIE studies, *31 Days Before Your CCNP and CCIE Enterprise Core Exam* most likely represents the first leg of your journey on your way to the destination: to become a Cisco Certified Network Professional or Cisco Certified Internetwork Expert. However, you might be reading this book at the *beginning* of your studies. If so, this book provides an excellent overview of the material you must now spend a great deal of time studying and practicing. But I must warn you: Unless you are extremely well versed in networking technologies and have considerable experience configuring and troubleshooting Cisco routers and switches, this book will *not* serve you well as the sole resource for your exam preparations. Therefore, let me spend some time discussing my recommendations for study resources.

Study Resources

Cisco Press and Pearson IT Certification offer an abundance of CCNP/CCIE-related books to serve as your primary source for learning how to implement core enterprise network technologies including dual-stack (IPv4 and IPv6) architecture, virtualization, infrastructure, network assurance, security, and automation.

Primary Resource

First on the list of important resources is the *CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide* by Jason Gooley, Ramiro Garza Rios, Bradley Edgeworth, and David Hucaby (ISBN: 978-1-58714-523-0). If you do not buy any other book, buy this one. With your purchase, you get access to practice exams and study materials and other online resources that make the price of the book very reasonable. There is no better resource on the market for a CCNP Enterprise and CCIE Enterprise Infrastructure or Wireless candidate.

Supplemental Resource

In addition to the book you hold in your hands, I recommend *CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide* (ISBN: 978-0-13-576816-7), which I co-wrote with my good friend and fellow Canadian Scott Empson. This book is much more than just a list of commands and what they do. Yes, it summarizes all the ENCOR and ENARSI IOS commands, keywords, command arguments, and associated prompts. In addition, it provides you with tips and examples of how to apply the commands to real-world scenarios. Configuration examples throughout the book provide you with a better understanding of how these commands are used in simple network designs.

The Cisco Learning Network

If you have not done so already, you should register with The Cisco Learning Network at https://learningnetwork.cisco.com. The Cisco Learning Network, sponsored by Cisco, is a free social learning network where IT professionals can engage in the common pursuit of enhancing and advancing their IT careers. Here you can find many resources to help you prepare for the ENCOR exam, in addition to a community of like-minded people ready to answer your questions, help you with your struggles, and share in your triumphs.

Cisco DevNet

For all things related to network programmability, check out https://developer.cisco.com. If you are looking to enhance or increase your skills with APIs, coding, Python, or even controller concepts, you can find a wealth of help at DevNet. At DevNet it is easy to find learning labs and content to help solidify current knowledge in network programmability. One of my personal favorites is the DevNet Sandbox (https://devnetsandbox.cisco.com/RM/Topology) where you can test drive and explore Cisco SD-Access, Cisco SD-WAN, Cisco DNA Center, Cisco Modeling Labs, Cisco Catalyst 9000, and Catalyst 9800 Wireless Controller labs.

Goals and Methods

The main goal of this book is to provide you with a clear and succinct review of information related to the CCNP/CCIE ENCOR exam objectives. Each day's exam topics are grouped into a common conceptual framework and use the following format:

- A title for the day that concisely states the overall topic
- A list of one or more ENCOR 350-401 exam topics to be reviewed
- A "Key Topics" section that introduces the review material and quickly orients you to the day's focus
- An extensive review section consisting of short paragraphs, lists, tables, examples, and graphics
- A "Study Resources" section to give you a quick reference for locating more in-depth treatment of the day's topics

The book counts down from Day 31 to Day 1, the last day before you take the exam. Inside this book are a calendar and checklist that you can tear out and use during your exam preparation.

Use the calendar to enter each actual date beside each countdown day and the exact day, time, and location of your ENCOR exam. The calendar provides a visual for the time you can dedicate to each ENCOR exam topic.

The checklist highlights important tasks and deadlines leading up to your exam. Use it to help map out your studies.

Who Should Read This Book?

The audience for this book is anyone in the final stages of preparing to take the ENCOR 350-401 exam. A secondary audience is anyone who needs a refresher review of ENCOR exam topics—possibly before attempting to recertify or sit for another certification for which the ENCOR exam is a prerequisite. For example, the ENCOR exam is now the qualifying exam for the CCIE Enterprise Infrastructure and CCIE Enterprise Wireless lab exam.

Getting to Know the ENCOR 350-401 Exam

For the current certification, announced in June 2019, Cisco created the ENCOR 350-401 exam. This book focuses on the entire list of topics published for that specific exam.

The ENCOR 350-401 exam is a 120-minute exam associated with the CCNP Enterprise, CCIE Enterprise Infrastructure, and CCIE Enterprise Wireless certifications. This exam tests a candidate's knowledge and skills related to implementing core enterprise network technologies, including dual-stack (IPv4 and IPv6) architecture, virtualization, infrastructure, network assurance, security, and automation.

Use the following steps to access a tutorial at home that demonstrates the exam environment before you go to take the exam:

Step 1. Visit http://learningnetwork.cisco.com.

Step 2. Search for "cisco certification exam tutorial".

Step 3. Look through the top results to find the page with videos that walk you through each exam question type.

As of April 2020, Cisco is allowing candidates to either take the certification exam at home or in a testing center. In both cases, the exams are still proctored by Pearson VUE.

For the online exam option, you need to perform a system check and install the OnVUE software on your PC. You must also have a reliable device with a webcam, a strong Internet connection, a quiet and private location, and government-issued identification.

For more information about online testing, see https://www.cisco.com/c/en/us/training-events/training-certifications/online-exam-proctoring.html.

If you decide to take the exam in a testing center, you first need to check in. The proctor verifies your identity, gives you some general instructions, and takes you into a quiet room containing a PC. When you're at the PC, you have some time before the timer starts on your exam. During this time, you can take the tutorial to get accustomed to the PC and the testing engine. Every time I sit for an exam, I go through the tutorial even though I know how the test engine works. It helps me settle my nerves and get focused. Anyone who has user-level skills in getting around a PC should have no problem with the testing environment.

When you start the exam, you are asked a series of questions, presented one at a time. You must answer each one before moving on to the next question. The exam engine does not let you go back and change any answers. Each exam question is in one of the following formats:

- **Multiple choice**: The multiple-choice format simply requires that you point and click a circle or check box next to the correct answer(s). Cisco traditionally tells you how many answers you need to choose, and the testing software prevents you from choosing too many or too few.

- **Fill in the blank**: Fill-in-the-blank questions usually require you to type only numbers. However, if words are requested, the case does not matter unless the answer is a command that is case sensitive (such as passwords and device names, when configuring authentication).

- **Drag and drop**: Drag-and-drop questions require you to click and hold, move a button or an icon to another area, and release the mouse button to place the object somewhere else—usually in a list. For some questions, to get the question correct, you might need to put a list of five things in the proper order.
- **Testlet**: A testlet contains one general scenario and several multiple-choice questions about the scenario. Testlets are ideal if you are confident in your knowledge of the scenario's content because you can leverage your strength over multiple questions.
- **Simlet**: A simlet is similar to a testlet, in that you are given a scenario with several multiple-choice questions. However, a simlet uses a network simulator to allow you access to a simulation of the command line of Cisco IOS Software. You can use **show** commands to examine a network's current behavior and answer the question.
- **Simulation**: A simulation also involves a network simulator, but you are given a task to accomplish, such as implementing a network solution or troubleshooting an existing network implementation. You do this by configuring one or more routers and switches. The exam grades the question based on the configuration you changed or added. A newer form of the simulation question is the GUI-based simulation, which simulates a graphical interface such as that found on a Cisco DNA Center or Cisco SD-WAN vManage dashboard.

Topics Covered on the CCNA Exam

Table I-1 summarizes the six domains of the ENCOR 350-401 exam.

Table I-1 ENCOR 350-401 Exam Domains and Weightings

Domain	Percentage of Exam
1.0 Architecture	15%
2.0 Virtualization	10%
3.0 Infrastructure	30%
4.0 Network Assurance	10%
5.0 Security	20%
6.0 Automation	15%

Although Cisco outlines general exam topics, some of these topics might not appear on the ENCOR exam; likewise, topics that are not specifically listed might appear on the exam. The exam topics that Cisco provides and that this book covers provide a general framework for exam preparation. Be sure to check Cisco's website for the latest exam topics: https://learningnetwork.cisco.com/s/encor-exam-topics.

Registering for the ENCOR 350-401 Exam

If you are starting this book 31 days before you take the ENCOR 350-401 exam, register for the exam right now. In my testing experience, there is no better motivator than a scheduled test date staring me in the face. I'm willing to bet the same holds true for you. Don't worry about unforeseen circumstances. You can cancel your exam registration for a full refund up to 24 hours before you are scheduled to take the exam. So, if you're ready, gather the following information and register right now!

- Legal name
- Social Security or passport number
- Company name
- Valid email address
- Method of payment

You can schedule your exam at any time by visiting www.pearsonvue.com/cisco/. I recommend that you schedule it for 31 days from now. The process and available test times vary based on the local testing center you choose.

Remember: There is no better motivation for study than an actual test date. *Sign up today*.

Day 31

Enterprise Network Architecture

ENCOR 350-401 Exam Topics

Architecture

- Explain the different design principles used in an enterprise network
  - Enterprise network design such as Tier 2, Tier 3, and Fabric Capacity planning

Key Topics

Today we review the hierarchical LAN design model, as well as the options available for different campus network deployments. This high-level overview of the enterprise campus architectures can be used to scale from a small corporate network environment to a large campus-sized network. We will look at design options such as:

- Two-tier design (collapsed core)
- Three-tier design
- The Layer 2 access layer (STP based)—both loop-free and looped
- The Layer 3 access layer (routed based)
- Simplified campus design using VSS and StackWise
- Software-Defined Access (SD-Access) design
- Spine-and-leaf architecture

Hierarchical LAN Design Model

The campus LAN uses a hierarchical design model to break up the design into modular groups or layers. Breaking up the design into layers allows each layer to implement specific functions, which simplifies the network design and therefore the deployment and management of the network.

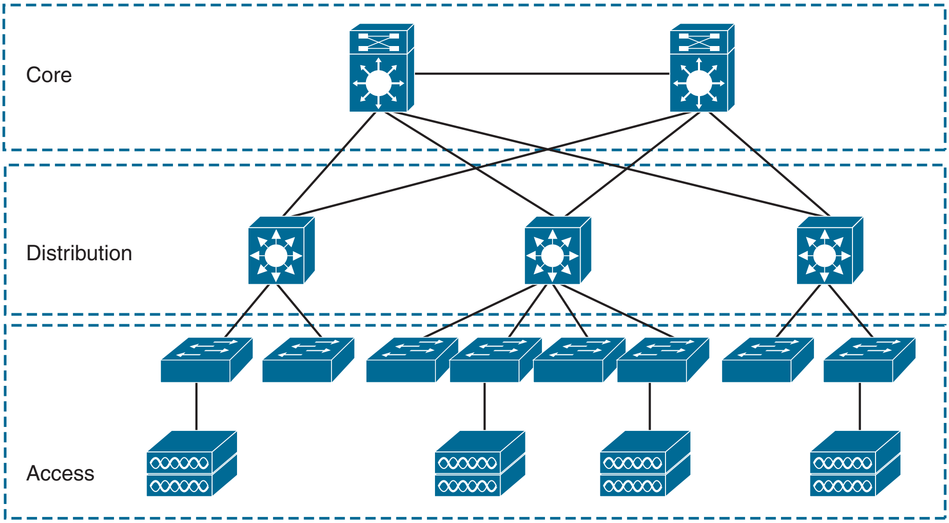
In flat or meshed network architectures, even small configuration changes tend to affect many systems. Hierarchical design helps constrain operational changes to a subset of the network, which makes the network easy to manage and improves resiliency. Modular structuring of the network into small, easy-to-understand elements also facilitates resiliency via improved fault isolation.

A hierarchical LAN design includes three layers:

- **Access layer:** Provides endpoints and users direct access to the network.
- **Distribution layer:** Aggregates access layers and provides connectivity to services.
- **Core layer:** Provides backbone connectivity between distribution layers for large LAN environments, as well as connectivity to other networks within or outside the organization.

Figure 31-1 illustrates a hierarchical LAN design using three layers.

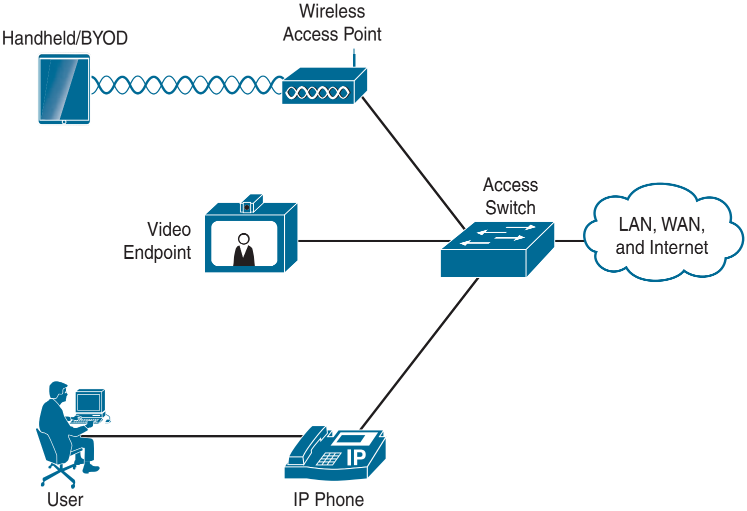
Figure 31-1 Hierarchical LAN Design



Access Layer

The access layer is where user-controlled devices, user-accessible devices, and other endpoint devices are connected to the network. The access layer provides both wired and wireless connectivity and contains features and services that ensure security and resiliency for the entire network. The access layer provides high-bandwidth device connectivity, as well as a set of network services that support advanced technologies, such as voice and video. The access layer—which provides a security, QoS, and policy trust boundary—is one of the most feature-rich parts of the campus network. It offers support for technologies like Power over Ethernet (PoE) and Cisco Discovery Protocol (CDP)/Link Layer Discovery Protocol (LLDP) for deployment of wireless access points (APs) and IP phones. Figure 31-2 illustrates connectivity at the access layer.

Figure 31-2 Access Layer Connectivity

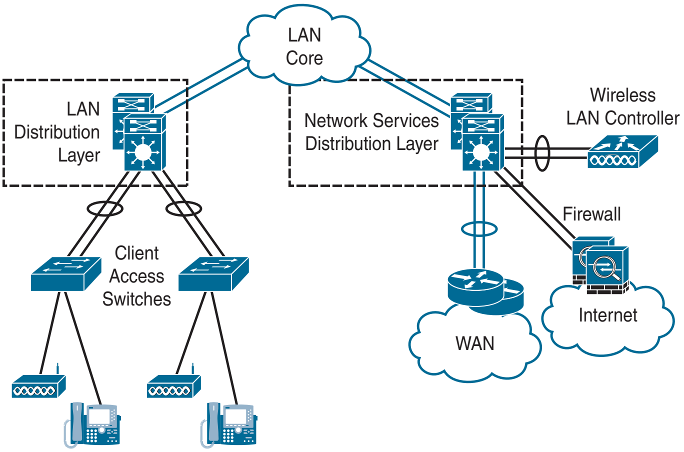

Handheld/BYOD
Wireless
Access Point
Access
Switch
LAN, WAN,
and Internet
Video
Endpoint
User
IP
IP Phone

Distribution Layer

In a network where connectivity needs to traverse the LAN end-to-end, whether between different access layer devices or from an access layer device to the WAN, the distribution layer facilitates this connectivity. The distribution layer provides scalability and resilience as it is used to logically aggregate the uplinks of access switches to one or more distribution switches. Scalability is accomplished via the aggregation of those access switches, and resilience is accomplished through the logical separation with multiple distribution switches. The distribution layer is the place where routing and packet manipulation are performed, and this layer can be a routing boundary between the access and core layers, where QoS and load balancing are implemented.

Figure 31-3 illustrates connectivity at the distribution layer.

Figure 31-3 Distribution Layer Connectivity

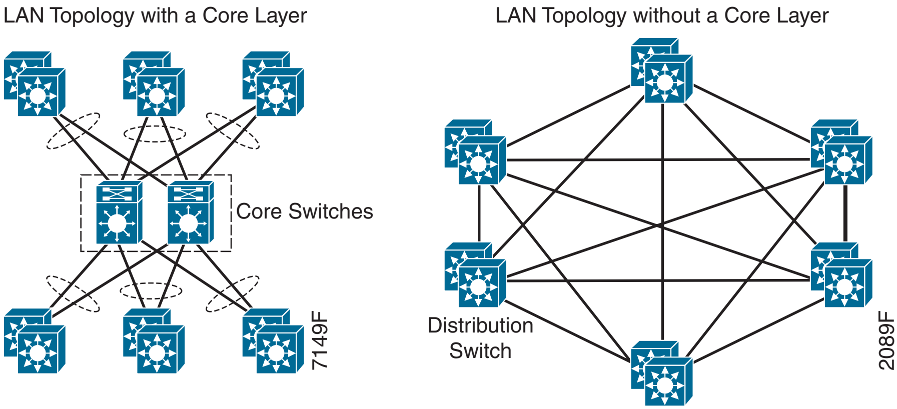

LAN
Core
LAN
Distribution
Layer
Network Services
Distribution Layer
Wireless
LAN Controller
Firewall
Client
Access
Switches
WAN
Internet

Core Layer

The core layer is the high-speed backbone for campus connectivity, and it is the aggregation point for the other layers and modules in the hierarchical network architecture. It is designed to switch packets with minimal processing as fast as possible 24x7x365. The core must provide a high level of stability, redundancy, and scalability. In environments where the campus is contained within a single building—or multiple adjacent buildings with the appropriate amount of fiber—it is possible to collapse the core into distribution switches. Without a core layer, the distribution layer switches need to be fully meshed. Such a design is difficult to scale and increases the cabling requirements because each new building distribution switch needs full-mesh connectivity to all the distribution switches. The routing complexity of a full-mesh design increases as you add new neighbors.

Figure 31-4 illustrates networks with and without a core layer. The core layer reduces the network complexity, from $N * (N - 1)$ to N links for N distributions if using link aggregation to the core, as shown in Figure 31-4; it would be $N * 2$ if using individual links to a redundant core.

Figure 31-4 LAN Topology With and Without a Core Layer


LAN Topology with a Core Layer
LAN Topology without a Core Layer
Core Switches
7149F
Distribution Switch
2089F

Enterprise Network Architecture Options

There are multiple enterprise network architecture design options available for deploying a campus network, depending on the size of the campus as well as the reliability, resiliency, availability, performance, security, and scalability required for it. Each possible option should be evaluated against business requirements. Since campus networks are modular, an enterprise network could have a mixture of these options.

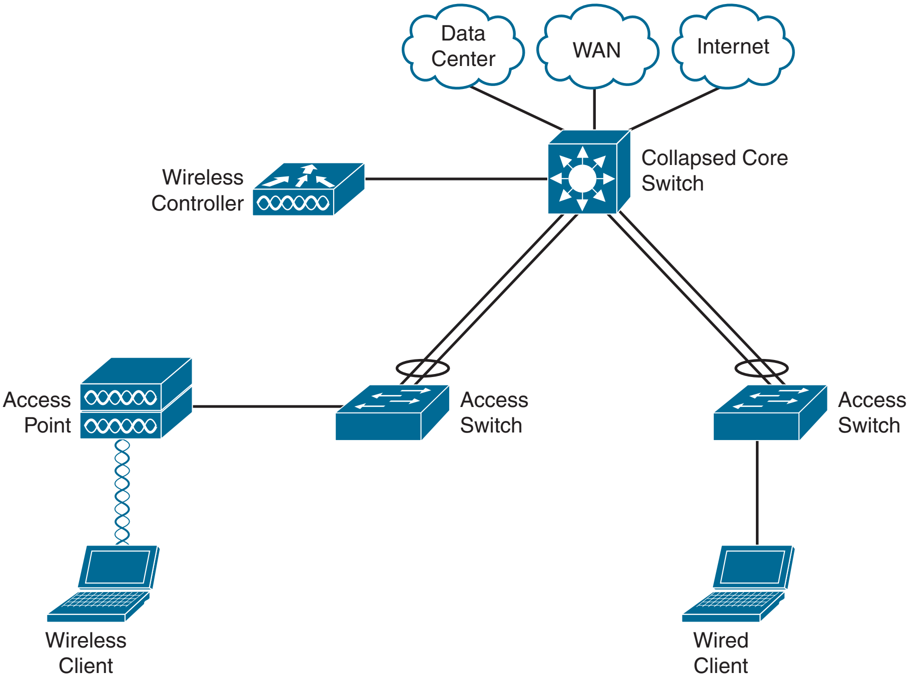
Two-Tier Design (Collapsed Core)

The distribution layer provides connectivity to network-based services, to the data center/server room, to the WAN, and to the Internet edge. Network-based services can include but are not limited to Cisco Identity Services Engine (ISE) and wireless LAN controllers (WLCs). Depending on the size of the LAN, these services and the interconnection to the WAN and Internet edge may reside on a distribution layer switch that also aggregates the LAN access layer connectivity. This is also referred to as a *collapsed core design* because the distribution serves as the Layer 3 aggregation layer for all devices.

It is important to consider that in any campus design—even designs that can be physically built with a collapsed core—the primary purpose of the core is to provide fault isolation and backbone connectivity. Isolating the distribution and core into two separate modules creates a clean delineation for change control between activities that affect end stations (laptops, phones, and printers) and activities that affect the data center, WAN, or other parts of the network. A core layer also provides for flexibility in adapting the campus design to meet physical cabling and geographic challenges.

Figure 31-5 illustrates a collapsed LAN core.

Figure 31-5 Two-Tier Design: Distribution Layer Functioning as a Collapsed Core

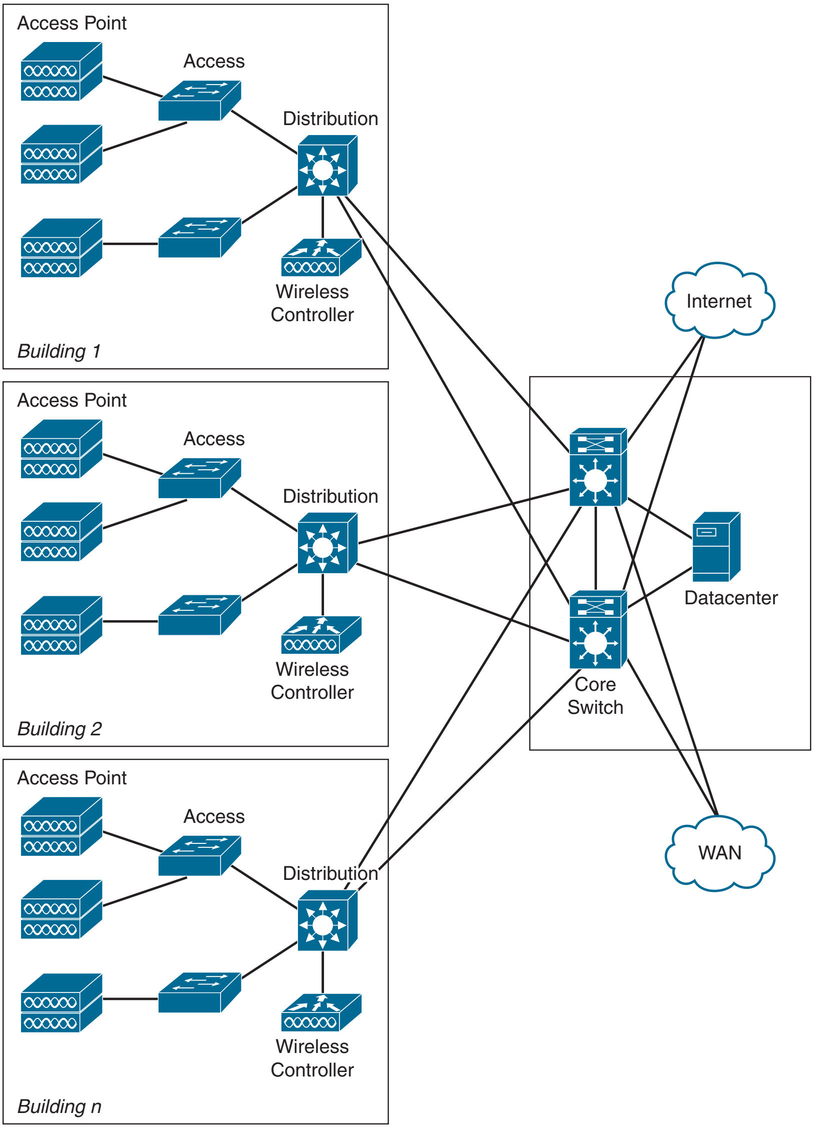


Three-Tier Design

In a large LAN, it would be difficult to share connectivity with access layer devices; therefore, a large LAN design may require a dedicated distribution layer for network-based services. As the density of WAN routers, Internet edge devices, and WLAN controllers grows, connections to individual distribution layer switches become hard to manage. When connecting at least three distribution layers together, using a core layer for distribution layer connectivity should be a consideration.

The three-tier campus network design is mostly deployed in environments where multiple offices and buildings are located closely together, allowing for high-speed fiber connections to the headquarters owned by the enterprise. Examples include the campus network at a university, a hospital with multiple buildings, and a large enterprise with multiple buildings on a privately owned campus. Figure 31-6 illustrates a typical three-tier campus network design.

Figure 31-6 Three-Tier Design for a Large Campus Network


Access Point
Access
Distribution
Wireless Controller
Building 1
Access Point
Access
Distribution
Wireless Controller
Building 2
Access Point
Access
Distribution
Wireless Controller
Building n
Internet
Datacenter
Core Switch
WAN

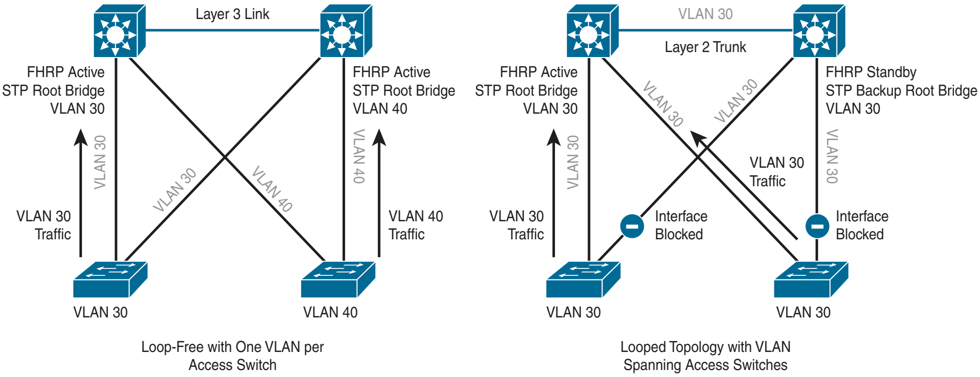
Layer 2 Access Layer (STP Based): Loop-Free and Looped

In the traditional hierarchical campus design, distribution blocks use a combination of Layer 2, Layer 3, and Layer 4 protocols and services to provide for optimal convergence, scalability, security, and manageability. In the most common distribution block configurations, the access switch is configured as a Layer 2 switch that forwards traffic on high-speed trunk ports to the distribution switches. Distribution switches are configured to support both Layer 2 switching on their downstream access switch trunks and Layer 3 switching on their upstream ports toward the core of the network.

With traditional Layer 2 access layer design, there is no true load balancing because STP blocks redundant links. Load balancing can be achieved through manipulation of STP and FHRP (HSRP, VRRP) settings and having traffic from different VLANs on different links. However, manual STP and FHRP manipulation is not true load balancing. Another way to achieve good load balancing is by limiting VLANs on a single switch and employing GLBP, but such a design could get complex. Convergence can also be an issue. Networks using RSTP have convergence times just below a second, but subsecond convergence is possible only with good hierarchical routing design and tuned FHRP settings and timers.

Figure 31-7 illustrates two Layer 2 access layer topologies: loop-free and looped. In a loop-free topology, a VLAN is constrained to a single switch, and a Layer 3 link is used between distribution layer switches to break the STP loop, ensuring that there are no blocked ports from the access layer to the distribution layer. In a looped topology, a VLAN spans multiple access switches. In this case, a Layer 2 trunk link is used between distribution layer switches. This design causes STP to block links, which reduces the bandwidth from the rest of the network and can cause slower network convergence.

Figure 31-7 Layer 2 Loop-Free and Looped Topologies


Layer 3 Link
FHRP Active
STP Root Bridge
VLAN 30
FHRP Active
STP Root Bridge
VLAN 40
VLAN 30
VLAN 30
VLAN 40
VLAN 40
VLAN 30
Traffic
VLAN 40
Traffic
VLAN 30
VLAN 40
Loop-Free with One VLAN per Access Switch
VLAN 30
Layer 2 Trunk
FHRP Active
STP Root Bridge
VLAN 30
FHRP Standby
STP Backup Root Bridge
VLAN 30
VLAN 30
VLAN 30
VLAN 30
VLAN 30
VLAN 30
Traffic
VLAN 30
Traffic
Interface
Blocked
Interface
Blocked
VLAN 30
VLAN 30
Looped Topology with VLAN Spanning Access Switches

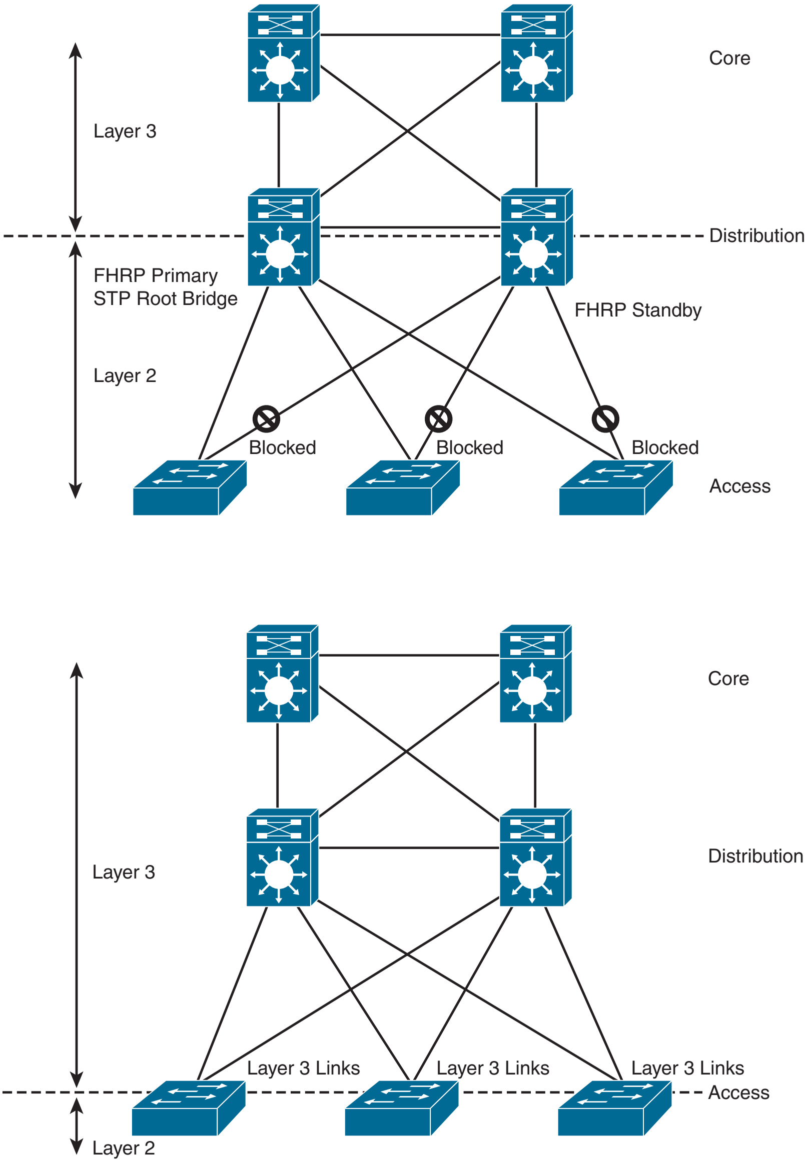
Layer 3 Access Layer (Routed Based)

An alternative configuration to the traditional distribution block model is one in which the access switch acts as a full Layer 3 routing node. The access-to-distribution Layer 2 uplink trunks are replaced with Layer 3 point-to-point routed links. This means the Layer 2/3 demarcation is moved from the distribution switch to the access switch. There is no need for FHRP, and every switch in the network participates in routing.

In both the traditional Layer 2 access layer and the Layer 3 routed access layer designs, each access switch is configured with unique voice and data VLANs. In the Layer 3 design, the default gateway and root bridge for these VLANs are simply moved from the distribution switch to the access switch. Addressing for all end stations and for the default gateway remains the same. VLAN and specific port configuration remains unchanged on the access switch. Router interface configuration, access lists, DHCP Helper, and other configurations for each VLAN remain identical. However, they are now configured on the VLAN SVI defined on the access switch instead of on the distribution switches. There are several notable configuration changes associated with the move of the Layer 3 interface down to the access switch. It is no longer necessary to configure an FHRP virtual gateway address as the "router" interfaces because all the VLANs are now local.

Figure 31-8 illustrates the difference between the traditional Layer 2 access layer design and the Layer 3 routed access layer design.

Figure 31-8 Layer 2 Access Layer and Layer 3 Access Layer Designs


Core
Layer 3
Distribution
FHRP Primary
STP Root Bridge
FHRP Standby
Layer 2
Blocked
Blocked
Blocked
Access
Core
Layer 3
Distribution
Layer 3 Links
Layer 3 Links
Layer 3 Links
Access
Layer 2

Simplified Campus Design Using VSS and StackWise

An alternative that can handle Layer 2 access layer requirements and avoid the complexity of the traditional multilayer campus is called a *simplified campus design*. This design uses multiple physical switches that act as a single logical switch, using either Virtual Switching System (VSS) or StackWise. One advantage of this design is that STP dependence is minimized, and all uplinks from the access layer to the distribution layer are active and forwarding traffic. Even the distributed VLAN design eliminates spanning tree blocked links caused by looped topologies. It is also possible to reduce dependence on spanning tree by using Multichassis EtherChannel (MEC) from the access layer with dual-homed uplinks. This is a key characteristic of this design, and load balancing between the physical distribution switches is possible because the access layer sees VSS as a single switch.

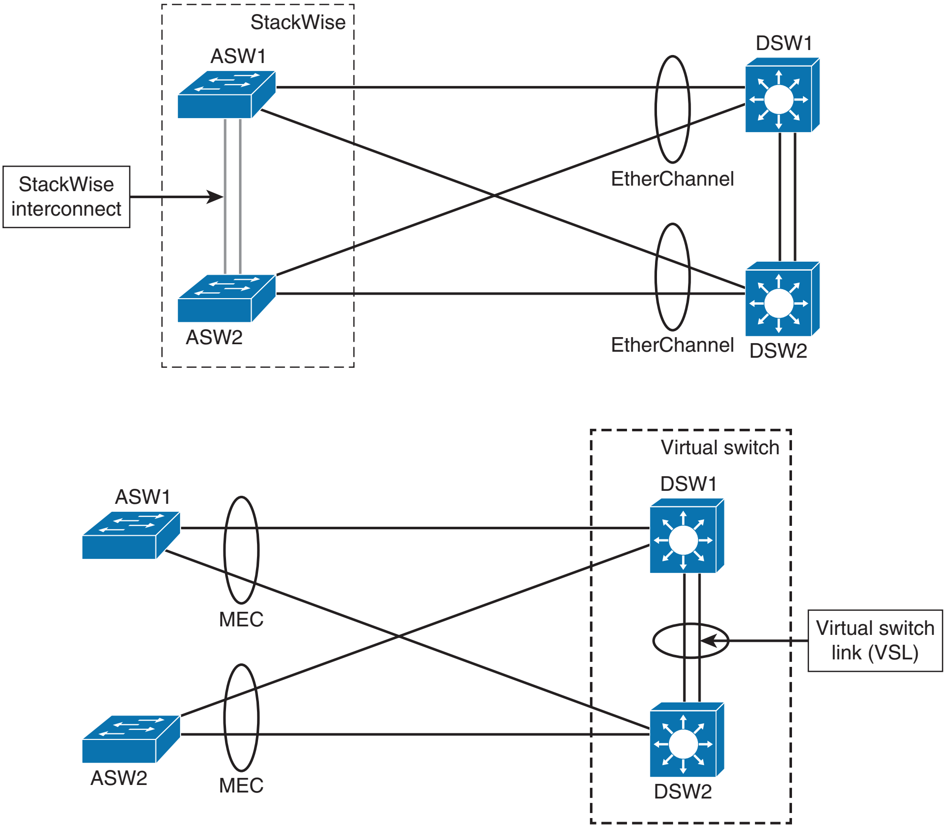
There are several other advantages to the simplified distribution layer design. Such a design does not need IP gateway redundancy protocols such as HSRP, VRRP, and GLBP because the default IP gateway is on a single logical interface, and resiliency is provided by the distribution layer VSS switch. Also, the network converges faster because it does not depend on spanning tree to unblock links when a failure occurs, thanks to MEC's fast subsecond failover between links in an uplink bundle.

Figure 31-9 illustrates the deployment of both StackWise and VSS technologies. In the top diagram, two access layer switches have been united into a single logical unit by using special stack interconnect cables that create a bidirectional closed-loop path. This bidirectional path acts as a switch fabric for all the connected switches. When a break is detected in a cable, the traffic is immediately wrapped back across the remaining path to continue forwarding. Also, in this scenario, the distribution layer switches are each configured with an EtherChannel link to the stacked access layer switches. This is possible because the two access layer switches are viewed as one logical switch from the perspective of the distribution layer.

In the bottom diagram, the two distribution layer switches have been configured as a VSS pair using a virtual switch link (VSL). The VSL is composed of up to eight 10 Gigabit Ethernet connections that are bundled into an EtherChannel. The VSL carries the control plane communication between the two VSS members, as well as regular user data traffic. Notice the use of MEC at the access layer. This allows the access layer switch to establish an EtherChannel to the two different physical chassis of the VSS pair. These links can be either Layer 2 trunks or Layer 3 routed connections.

Keep in mind that it is possible to combine StackWise and VSS in a campus network. They are not mutually exclusive. StackWise is typically found at the access layer, whereas VSS is found at the distribution and core layers.

Figure 31-9 Simplified Campus Design with VSS and StackWise


StackWise
ASW1
DSW1
StackWise interconnect
EtherChannel
EtherChannel
ASW2
DSW2
Virtual switch
DSW1
ASW1
MEC
Virtual switch link (VSL)
ASW2
MEC
DSW2

Common Access–Distribution Interconnection Designs

To summarize, there are four common access–distribution interconnection design options:

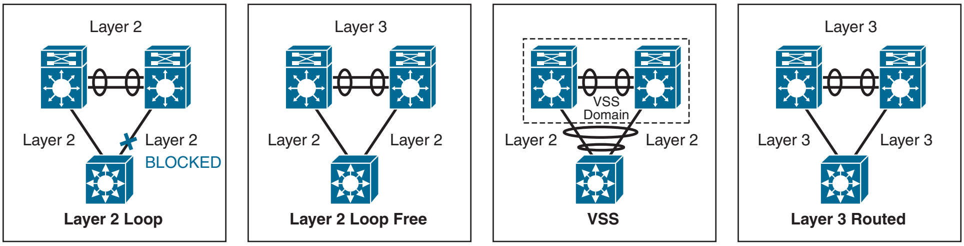
- **Layer 2 looped design:** Uses Layer 2 switching at the access layer and on the distribution switch interconnect. This introduces a Layer 2 loop between distribution switches and access switches. STP blocks one of the uplinks from the access switch to the distribution switches. The reconvergence time in the event of uplink failure depends on STP and FHRP convergence times.

- **Layer 2 loop-free design:** Uses Layer 2 switching at the access layer and Layer 3 routing on the distribution switch interconnect. There are no Layer 2 loops between the access switch and the distribution switches. Both uplinks from the access layer switch are forwarding. Reconvergence time in the event of an uplink failure depends on the FHRP convergence time.

- **VSS design:** Results in STP recognizing an EtherChannel link as a single logical link. STP is thus effectively removed from the access–distribution block. STP is needed only on access switch ports that connect to end devices to protect against end-user-created loops. If one of the links between access and distribution switches fails, forwarding of traffic continues without the need for reconvergence.
- **Layer 3 routed design:** Uses Layer 3 routing on the access switches and the distribution switch interconnect. There are no Layer 2 loops between the access layer switch and distribution layer switches. The need for STP is eliminated, except on connections from the access layer switch to end devices to protect against end-user wiring errors. Reconvergence time in the event of uplink failure depends solely on the routing protocol convergence times.

Figure 31-10 illustrates the four access–distribution interconnection design options.

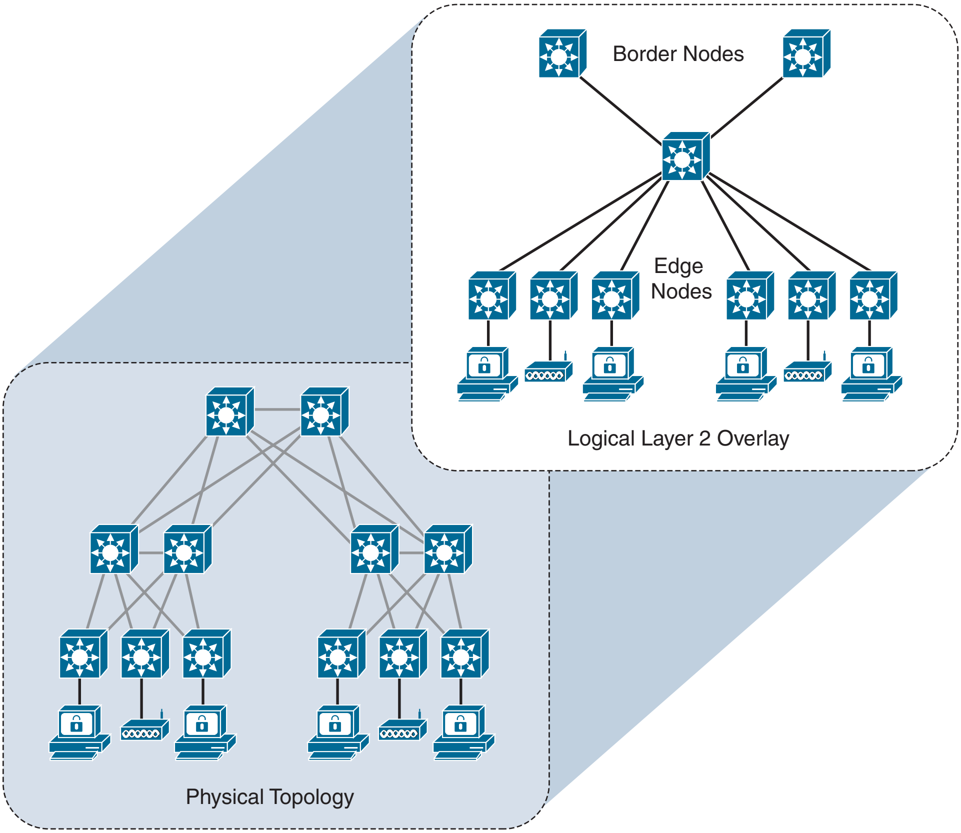
Figure 31-10 Access–Distribution Interconnection Design Options



Software-Defined Access (SD-Access) Design

You can overcome the Layer 2 limitations of the routed access layer design by adding fabric capability to a campus network that is already using a Layer 3 access network; the addition of the fabric is automated using SD-Access technology. The SD-Access design enables the use of virtual networks (called *overlay networks*) running on a physical network (called the *underlay network*) in order to create alternative topologies to connect devices. In addition to network virtualization, SD-Access allows for software-defined segmentation and policy enforcement based on user identity and group membership, integrated with Cisco TrustSec technology. Figure 31-11 illustrates the relationship between the physical underlay network and the Layer 2 virtual overlay network used in SD-Access environments. SD-Access is covered in more detail on Day 18.

Figure 31-11 Layer 2 SD-Access Overlay

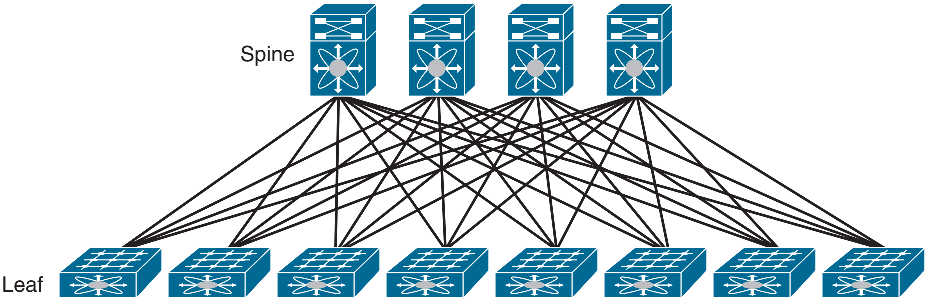


Spine-and-Leaf Architecture

A new data center design called the *Clos* network–based spine-and-leaf architecture was developed to overcome limitations such as server-to-server latency and bandwidth bottlenecks typically found in three-tier data center architectures. This new architecture has been proven to deliver the high-bandwidth, low-latency, nonblocking server-to-server connectivity needed to support high-speed workloads and shift the focus from earlier 1 Gbps or 10 Gbps uplinks to the modern 100 Gbps uplinks necessary in today's data centers. Figure 31-12 illustrates a typical two-tiered spine-and-leaf topology.

In this two-tier Clos architecture, every lower-tier switch (leaf layer) is connected to each of the top-tier switches (spine layer) in a full-mesh topology. The leaf layer consists of access switches that connect to devices such as servers. The spine layer, which is the backbone of the network, is responsible for interconnecting all leaf switches. Every leaf switch connects to every spine switch in the fabric. A path through the network is randomly chosen so that the traffic load is evenly distributed among the top-tier switches. If one of the top-tier switches were to fail, performance throughout the data center would degrade only slightly.

Figure 31-12 Typical Spine-and-Leaf Topology



If oversubscription of a link occurs (that is, if more traffic is generated than can be aggregated on the active link at one time), the process for expanding capacity is straightforward: An additional spine switch can be added, and uplinks can be extended to every leaf switch, resulting in the addition of interlayer bandwidth and reduction of the oversubscription. If device port capacity becomes a concern, it is possible to add a new leaf switch by connecting it to every spine switch and adding the network configuration to the switch. The ease of expansion optimizes the IT department's process of scaling the network. If no oversubscription occurs between the lower-tier switches and their uplinks, a nonblocking architecture can be achieved.

With a spine-and-leaf architecture, no matter which leaf switches are connected to servers, the traffic always has to cross the same number of devices to get to another server (unless the other server is located on the same leaf). This approach keeps latency at a predictable level because a payload only has to hop to a spine switch and another leaf switch to reach its destination.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	22
CCNP Enterprise Design ENSLD 300-420 Official Cert Guide: Designing Cisco Enterprise Networks	6, 7
Transforming Campus Networks to Intent-Based Networking	1
Campus LAN and Wireless LAN Design Guide	https://www.cisco.com/c/en/us/td/docs/solutions/CVD/Campus/cisco-campus-lan-wlan-design-guide.html
Cisco Data Center Spine-and-Leaf Architecture: Design Overview White Paper	https://www.cisco.com/c/en/us/products/collateral/switches/nexus-7000-series-switches/white-paper-c11-737022.html

This page intentionally left blank

Day 30

Packet Switching and Forwarding

ENCOR 350-401 Exam Topics

Architecture

- Differentiate hardware and software switching mechanisms
    - Process and CEF
    - MAC address table and TCAM
    - sFIB vs. RIB

Key Topics

Today we review the information bases that are used in routing, such as the Forwarding Information Base (FIB) and Routing Information Base (RIB), as well as the two types of memory tables used in switching: content-addressable memory (CAM) and ternary content-addressable memory (TCAM). We also review different software and hardware switching mechanisms, such as process switching, fast switching, and Cisco Express Forwarding (CEF). Finally, we examine switch hardware redundancy mechanisms such as Stateful Switchover (SSO) and Nonstop Forwarding (NSF), and look at how switches use Switching Database Manager (SDM) templates to allocate internal resources.

Layer 2 Switch Operation

An Ethernet switch operates at Layer 2 of the Open Systems Interconnection (OSI) model. The switch makes decisions about forwarding frames that are based on the destination Media Access Control (MAC) address in the frame. To figure out where a frame must be sent, the switch looks in its MAC address table. This information can be told to the switch, or the switch can learn it automatically. The switch listens to incoming frames and checks the source MAC addresses. If the address is not in the table already, the MAC address, switch port, and VLAN are recorded in the forwarding table. The forwarding table is also called the content-addressable memory (CAM) table. Note that if the destination MAC address of the frame is unknown, the switch forwards the frame through all ports within a virtual local-area network (VLAN). This behavior is known as *unknown unicast flooding*. Broadcast and multicast traffic is destined for multiple destinations, so they are also flooded by default.

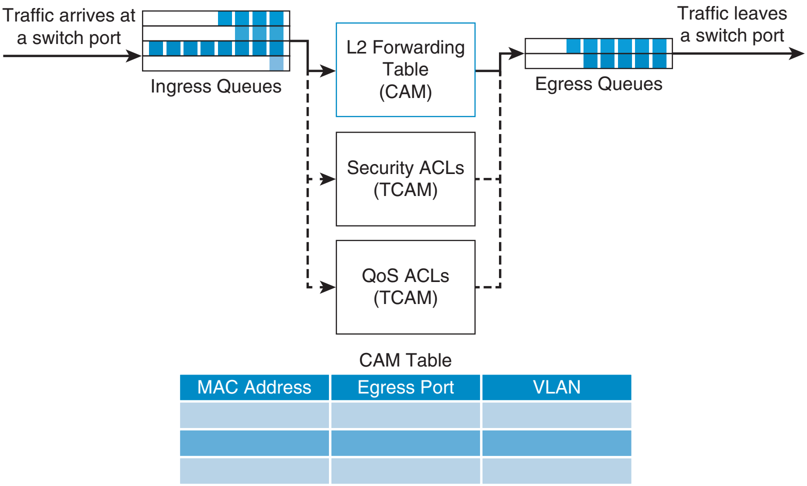
Table 30-1 shows a typical Layer 2 switch CAM table. If the switch receives a frame on port 1 and the destination MAC address for the frame is 0000.0000.3333, the switch looks up its forwarding table and figures out that MAC address 0000.0000.3333 is recorded on port 5. The switch forwards the frame through port 5. If, instead, the switch receives a broadcast frame on port 1, the switch forwards the frame through all ports that are within the same VLAN. The frame was received on port 1, which is in VLAN 1; therefore, the frame is forwarded through all ports on the switch that belong to VLAN 1 (all ports except port 3).

Table 30-1 Sample CAM Table in a Switch

MAC Address	Port	VLAN
0000.0000.1111	1	1
0000.0000.2222	2	1
0000.0000.6666	6	1
0000.0000.3333	5	1
0000.0000.8888	3	20
0000.0000.AAAA	4	1

When a switch receives a frame, it places the frame into a port ingress queue. Figure 30-1 illustrates this process. A port can have multiple ingress queues, and typically these queues have different priorities. Important frames are processed sooner than less important ones.

Figure 30-1 Layer 2 Traffic Switching Process


Traffic arrives at a switch port
Ingress Queues
L2 Forwarding Table (CAM)
Security ACLs (TCAM)
QoS ACLs (TCAM)
Egress Queues
Traffic leaves a switch port
CAM Table
MAC Address
Egress Port
VLAN

When the switch selects a frame from the queue, it needs to answer a few questions:

- Where should I forward the frame?
- Should I even forward the frame?
- How should I forward the frame?

Decisions about these three questions are answered based on the following:

- **Layer 2 forwarding table:** MAC addresses in the CAM table are used as indexes. If the MAC address of an incoming frame is found in the CAM table, the frame is forwarded through the MAC-bound port. If the address is not found, the frame is flooded through all ports in the VLAN.

- **Access control lists (ACLs):** ACLs can identify a frame according to its MAC addresses. The ternary content-addressable memory (TCAM) contains these ACLs. A single lookup is needed to decide whether the frame should be forwarded.
- **Quality of service (QoS):** Incoming frames can be classified according to QoS parameters. Traffic can then be prioritized and rate limited. QoS decisions are also made by TCAM in a single table lookup.

After CAM and TCAM table lookups are done, the frame is placed into an egress queue on the appropriate outbound switch port. The appropriate egress queue is determined by QoS, and the important frames are processed first.

MAC Address Table and TCAM

Cisco switches maintain CAM and TCAM tables. CAM is used in Layer 2 switching, and TCAM is used in Layer 3 switching. Both tables are kept in fast memory so that data processing occurs quickly.

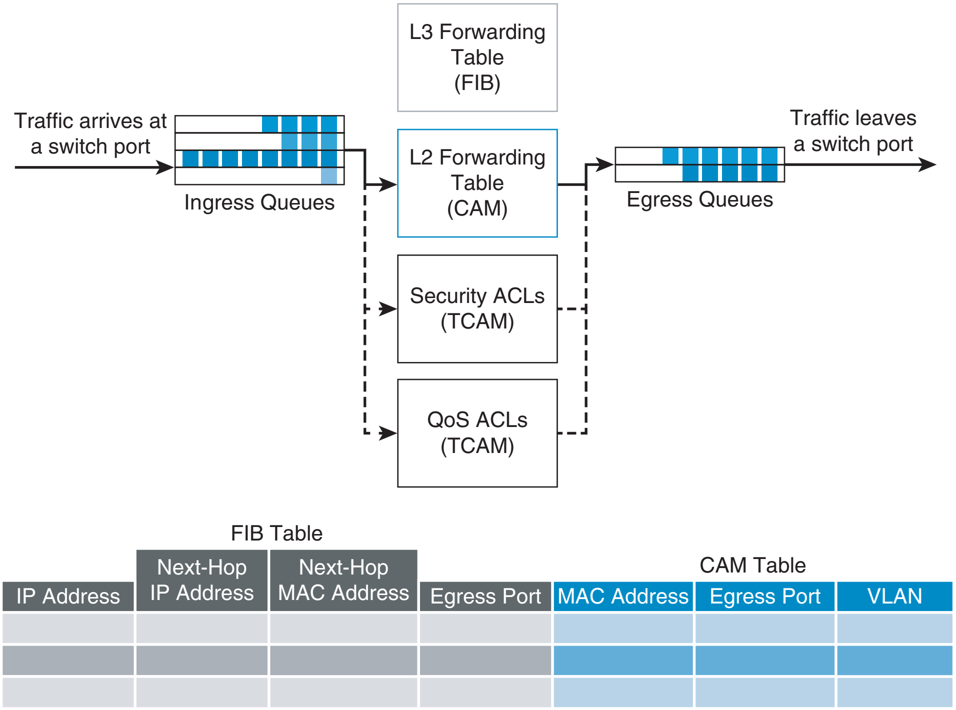
Multilayer switches forward frames and packets at wire speed by using ASIC hardware. Specific Layer 2 and Layer 3 components, such as learned MAC addresses or ACLs, are cached into the hardware. These tables are stored in CAM and TCAM.

- **CAM table:** The CAM table is the primary table that is used to make Layer 2 forwarding decisions. The table is built by recording the source MAC address and inbound port number for each incoming frame.
- **TCAM table:** The TCAM table stores ACL, QoS, and other information that is generally associated with upper-layer processing. Most switches have multiple TCAMs, such as one for inbound ACLs, one for outbound ACLs, one for QoS, and so on. Multiple TCAMs allow switches to perform different checks in parallel, thus shortening the packet-processing time. Cisco switches perform CAM and TCAM lookups in parallel. TCAM uses a table-lookup operation that is greatly enhanced to allow a more abstract operation than is possible with CAM. For example, binary values (0s and 1s) make up a key in the table, but a mask value is also used to decide which bits of the key are relevant. This effectively makes a key consisting of three input values: 0, 1, and X (do not care) bit values—a threefold, or *ternary*, combination. TCAM entries are composed of value, mask, and result (VMR) combinations. Fields from frame or packet headers are fed into TCAM, where they are matched against the value and mask pairs to yield a result. For example, for an ACL entry, the value and mask fields would contain the source and destination IP addresses being matched as well as the wildcard mask that indicates the number of bits to match. The result would either be "permit" or "deny," depending on the access control entry (ACE) being checked.

Layer 3 Switch Operation

Multilayer switches not only perform Layer 2 switching but also forward frames that are based on Layer 3 and Layer 4 information. Multilayer switches combine the functions of a switch and a router, and they also have a flow cache component. Figure 30-2 illustrates what occurs when a packet is pulled off an ingress queue, and the switch inspects the Layer 2 and Layer 3 destination addresses.

Figure 30-2 Layer 3 Traffic Switching Process


L3 Forwarding Table (FIB)
Traffic arrives at a switch port
Ingress Queues
L2 Forwarding Table (CAM)
Traffic leaves a switch port
Egress Queues
Security ACLs (TCAM)
QoS ACLs (TCAM)
FIB Table
IP Address
Next-Hop IP Address
Next-Hop MAC Address
Egress Port
CAM Table
MAC Address
Egress Port
VLAN

As with a Layer 2 switch, a Layer 3 switch needs to answer a few questions:

- Where should I forward the frame?
- Should I even forward the frame?
- How should I forward the frame?

Decisions about these three questions are made as follows:

- **Layer 2 forwarding table:** MAC addresses in the CAM table are used as indexes. If the frame encapsulates a Layer 3 packet that needs to be routed, the destination MAC address of the frame is that of the Layer 3 interface on the switch for that VLAN.
- **Layer 3 forwarding table:** The IP addresses in the FIB table are used as indexes. The best match to the destination IP address is the Layer 3 next-hop address. The FIB also lists next-hop MAC addresses, the egress switch port, and the VLAN ID, so there is no need for additional lookup.
- **ACLs:** The TCAM contains ACLs. A single lookup is needed to decide whether the frame should be forwarded.
- **QoS:** Incoming frames can be classified according to QoS parameters. Traffic can then be prioritized and rate limited. QoS decisions are also made by the TCAM in a single table lookup.

After CAM and TCAM table lookups are done, the packet is placed into an egress queue on the appropriate outbound switch port. The appropriate egress queue is determined by QoS, and the important packets are processed first.

Forwarding Mechanisms

Packet forwarding is a core router function; therefore, high-speed packet forwarding is very important. Throughout the years, various methods of packet switching have been developed. Cisco IOS platform-switching mechanisms evolved from process switching to fast switching and eventually to CEF switching.

Control and Data Plane

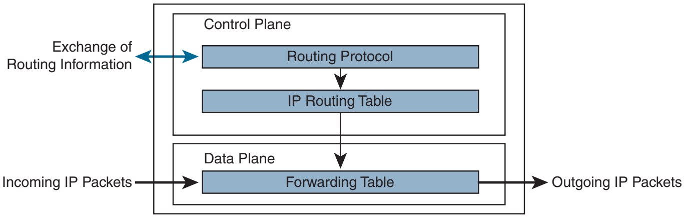
A network device has three planes of operation: the management plane, the control plane, and the forwarding plane. A Layer 3 device employs a distributed architecture in which the control plane and data plane are relatively independent. For example, the exchange of routing protocol information is performed in the control plane by the route processor, whereas data packets are forwarded in the data plane by an interface microcoded processor.

The main functions of the control layer between the routing protocol and the firmware data plane microcode include the following:

- Managing the internal data and control circuits for the packet-forwarding and control functions
- Extracting the other routing and packet-forwarding-related control information from Layer 2 and Layer 3 bridging and routing protocols and the configuration data and then conveying the information to the interface module for control of the data plane
- Collecting the data plane information, such as traffic statistics, from the interface module to the route processor (RP)
- Handling certain data packets that are sent from the Ethernet interface modules to the route processor

Figure 30-3 illustrates the relationship between the control plane and the data plane. In the diagram, the router's routing protocol builds the routing table using information it gathers from and exchanges with its neighbors. The router builds a forwarding table in the data plane to process incoming packets.

Figure 30-3 Control and Data Plane Operations

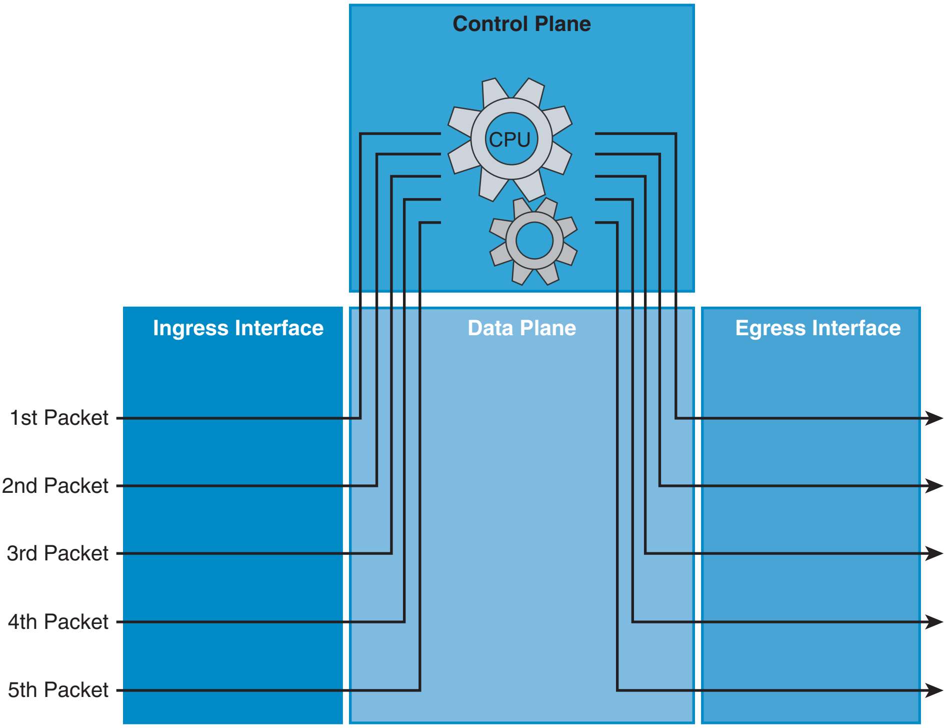


Cisco Switching Mechanisms

Cisco routers support three switching mechanisms that are used to make forwarding decisions:

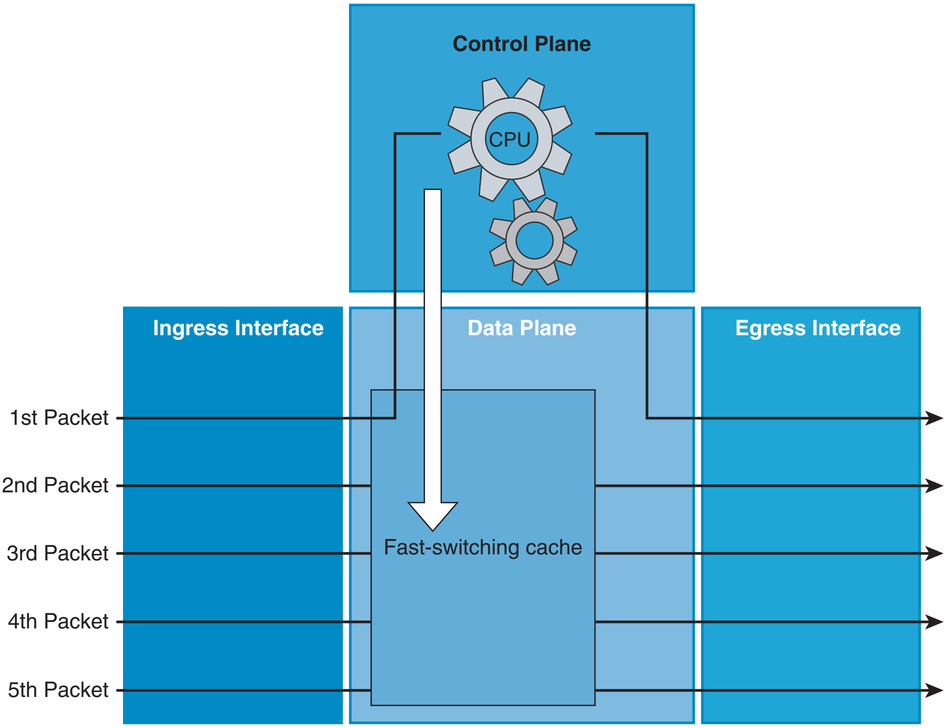
- **Process switching:** In this switching method, the router strips off the Layer 2 header for each incoming frame, looks up the Layer 3 destination network address in the routing table for each packet, and sends the frame with the rewritten Layer 2 header, including a computed cyclic redundancy check (CRC) to the outgoing interface. All these operations are done by software that is running on the CPU for each individual frame. Process switching is the most CPU-intensive method available in Cisco routers. It greatly degrades performance and is generally used only as a last resort or during troubleshooting. Figure 30-4 illustrates this type of switching.

Figure 30-4 Process-Switched Packets



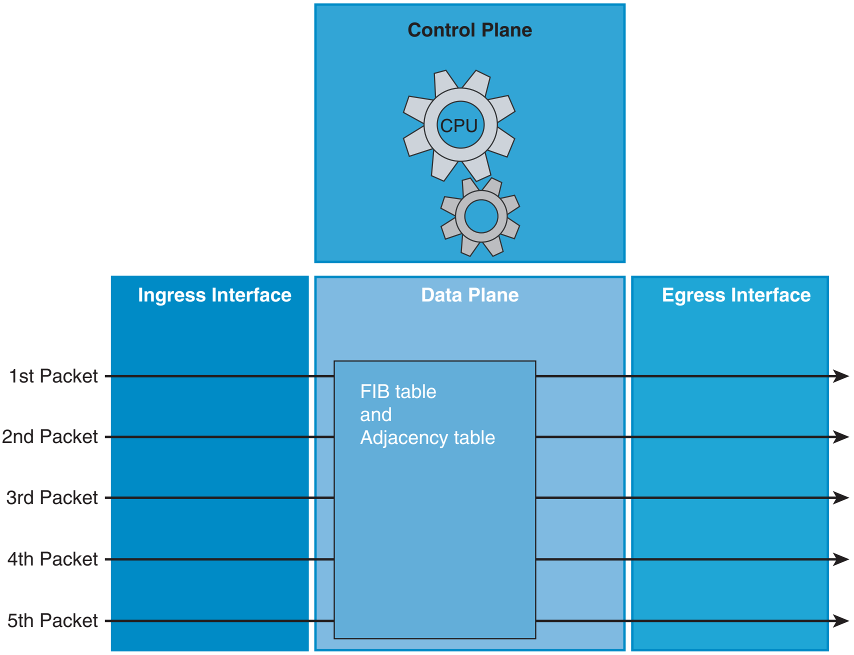
- **Fast Switching:** This switching method is faster than process switching. With fast switching, the initial packet of a traffic flow is process switched. This means that it is examined by the CPU, and the forwarding decision is made in software. However, the forwarding decision is also stored in the data plane hardware fast-switching cache. When subsequent frames in the flow arrive, the destination is found in the hardware fast-switching cache, and the frames are then forwarded without interrupting the CPU. Figure 30-5 illustrates how only the first packet of a flow is process switched and added to the fast-switching cache. The next four packets are quickly processed based on the information in the fast-switching cache; the initial packet of a traffic flow is process switched. On a Layer 3 switch, fast switching is also called route caching, flow-based switching, or demand-based switching. Route caching means that when the switch detects a traffic flow into the switch, a Layer 3 route cache is built within hardware functions.

Figure 30-5 Fast-Switched Packets



- **Cisco Express Forwarding (CEF):** This switching method is the fastest switching mode and is less CPU intensive than fast switching and process switching. The control plane CPU of a CEF-enabled router creates two hardware-based tables called the Forwarding Information Base (FIB) table and an adjacency table using the Layer 3 routing table as well as a Layer 2 Address Resolution Protocol (ARP) table. When a network has converged, the FIB and adjacency tables contain all the information a router needs when forwarding a packet. As illustrated in Figure 30-6, these two tables are then used to make hardware-based forwarding decisions for all frames in a data flow—even the first frame. The FIB contains precomputed reverse lookups and next-hop information for routes, including the interface and Layer 2 information. While CEF is the fastest switching mode, there are limitations. Some features are not compatible with CEF. There are also some rare instances in which the CEF can actually degrade performance. A typical case of such degradation is called *CEF polarization*. This is found in a topology that uses load-balanced Layer 3 paths but where only one path per given host pair is constantly used. Packets that cannot be CEF switched, such as packets destined to the router itself, are "punted," and the packet is fast switched or process switched. On a Layer 3 switch, CEF is also called topology-based switching. Information from the routing table is used to populate the route cache, regardless of traffic flow. The populated route cache is the FIB, and CEF is the facility that builds the FIB.

Figure 30-6 CEF-Switched Packets

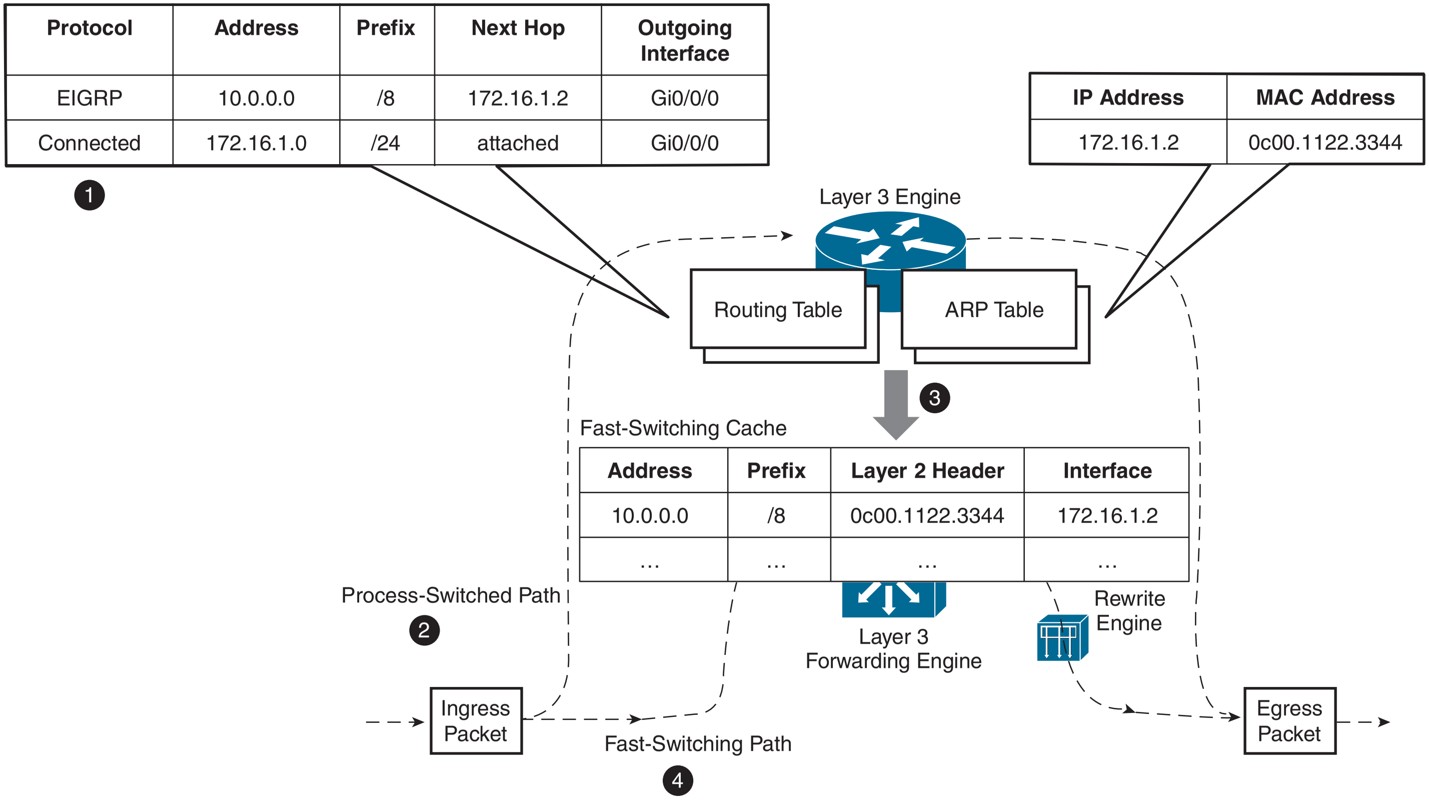

Control Plane
CPU
Ingress Interface
Data Plane
Egress Interface
1st Packet
2nd Packet
3rd Packet
4th Packet
5th Packet
FIB table
and
Adjacency table

Process and Fast Switching

A specific sequence of events occurs when process switching and fast switching are used for destinations that were learned through a routing protocol such as Cisco's Enhanced Interior Gateway Routing Protocol (EIGRP). Figure 30-7 demonstrates this process, which illustrates the following steps:

1. When an EIGRP update is received and processed, an entry is created in the routing table.
2. When the first packet arrives for this destination, the router tries to find the destination in the fast-switching cache. Because the destination is not in the fast-switching cache, process switching must switch the packet when the process is run. The process performs a recursive lookup to find the outgoing interface. The process switching might trigger an ARP request or find the Layer 2 address in the ARP cache.
3. The router creates an entry in the fast-switching cache.
4. All subsequent packets for the same destination are fast switched:
    - The switching occurs in the interrupt code. (The packet is processed immediately.)
    - Fast destination lookup is performed (no recursion).
    - The encapsulation uses a pre-generated Layer 2 header that contains the destination and Layer 2 source MAC address. (No ARP request or ARP cache lookup is necessary.)

Figure 30-7 Process- and Fast-Switching Example



Protocol	Address	Prefix	Next Hop	Outgoing Interface
EIGRP	10.0.0.0	/8	172.16.1.2	Gi0/0/0
Connected	172.16.1.0	/24	attached	Gi0/0/0

IP Address	MAC Address
172.16.1.2	0c00.1122.3344

Fast-Switching Cache

Address	Prefix	Layer 2 Header	Interface
10.0.0.0	/8	0c00.1122.3344	172.16.1.2
…	…	…	…

❶
Layer 3 Engine
Routing Table
ARP Table
❸
Process-Switched Path
❷
Layer 3
Forwarding Engine
Rewrite
Engine
Ingress
Packet
Fast-Switching Path
❹
Egress
Packet

> Whenever a router receives a packet that should be fast switched but the destination is not in the switching cache, the packet is process switched. A full routing table lookup is performed, and an entry in the fast-switching cache is created to ensure that the subsequent packets for the same destination prefix will be fast switched.

Cisco Express Forwarding

Cisco Express Forwarding uses special strategies to switch data packets to their destinations. It caches the information that is generated by the Layer 3 routing engine even before the router encounters any data flows. Cisco Express Forwarding caches routing information in one table (the FIB) and caches Layer 2 next-hop addresses and frame header rewrite information for all FIB entries in another table, called the adjacency table.

Figure 30-8 illustrates how CEF switching operates.

Cisco Express Forwarding separates the control plane software from the data plane hardware to achieve higher data throughput. The control plane is responsible for building the FIB table and adjacency tables in software. The data plane is responsible for forwarding IP unicast traffic using hardware.

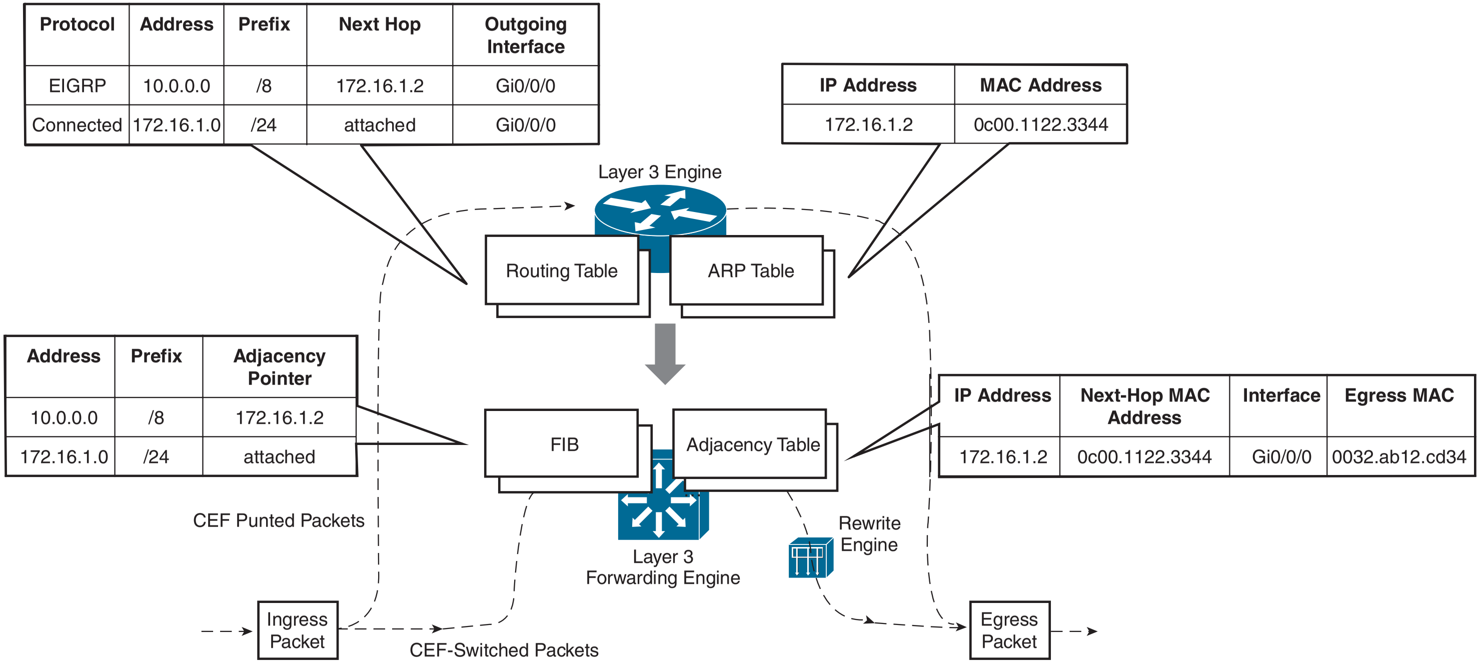
Routing protocols such as OSPF, EIGRP, and BGP each have their own Routing Information Base (RIB). From individual routing protocol RIBs, the best routes to each destination network are selected to install in the global RIB or the IP routing table.

The FIB is derived from the IP routing table and is arranged for maximum lookup throughput. CEF IP destination prefixes are stored in the TCAM table, from the most-specific entry to the least-specific entry. The FIB lookup is based on the Layer 3 destination address prefix (longest match), so it matches the structure of CEF entries within the TCAM. When the CEF TCAM table is full, a wildcard entry redirects frames to the Layer 3 engine. The FIB table is updated after each network change, but only once, and it contains all known routes; there is no need to build a route cache by centrally processing initial packets from each data flow. Each change in the IP routing table triggers a similar change in the FIB table because it contains all next-hop addresses that are associated with all destination networks.

The adjacency table is derived from the ARP table, and it contains Layer 2 header rewrite (MAC) information for each next hop that is contained in the FIB. Nodes in the network are said to be adjacent if they are within a single hop of each other. The adjacency table maintains Layer 2 next-hop addresses and link-layer header information for all FIB entries. The adjacency table is populated as adjacencies are discovered. Each time an adjacency entry is created (such as through ARP), a link-layer header for that adjacent node is precomputed and is stored in the adjacency table. When the adjacency table is full, a CEF TCAM table entry points to the Layer 3 engine to redirect the adjacency.

The rewrite engine is responsible for building the new frame's source and destination MAC addresses, decrementing the Time-to-Live (TTL) field, recomputing a new IP header checksum, and forwarding the packet to the next-hop device.

Figure 30-8 CEF Switching Example



Protocol	Address	Prefix	Next Hop	Outgoing Interface
EIGRP	10.0.0.0	/8	172.16.1.2	Gi0/0/0
Connected	172.16.1.0	/24	attached	Gi0/0/0

IP Address	MAC Address
172.16.1.2	0c00.1122.3344

Address	Prefix	Adjacency Pointer
10.0.0.0	/8	172.16.1.2
172.16.1.0	/24	attached

IP Address	Next-Hop MAC Address	Interface	Egress MAC
172.16.1.2	0c00.1122.3344	Gi0/0/0	0032.ab12.cd34

Layer 3 Engine
Routing Table
ARP Table
FIB
Adjacency Table
Layer 3
Forwarding Engine
Rewrite
Engine
CEF Punted Packets
Ingress
Packet
CEF-Switched Packets
Egress
Packet

Not all packets can be processed in the hardware. When traffic cannot be processed in the hardware, it must be received by software processing of the Layer 3 engine. This traffic does not receive the benefit of expedited hardware-based forwarding. Several different packet types may force the Layer 3 engine to process them.

IP exception packets, or "punts," have the following characteristics:

- They use IP header options.
- They have an expiring IP TTL counter.
- They are forwarded to a tunnel interface.
- They arrive with unsupported encapsulation types.
- They are routed to an interface with unsupported encapsulation types.
- They exceed the Maximum Transmission Unit (MTU) value of an output interface and must be fragmented.

Centralized and Distributed Switching

Layer 3 CEF switching can occur at two different locations on a switch:

- **Centralized switching:** With this type, switching decisions are made on the route processor by a central forwarding table, typically controlled by an ASIC. When centralized CEF is enabled, the CEF FIB and adjacency tables reside on the RP, and the RP performs the CEF forwarding. Figure 30-9 shows the relationship between the routing table, the FIB, and the adjacency table in central Cisco Express Forwarding mode operation. Traffic is forwarded between LANs to a device on the enterprise network that is running central CEF. The RP performs the CEF forwarding.
- **Distributed switching (dCEF):** Switching decisions can be made on a port or at line-card level rather than on a central route processor. Cached tables are distributed and synchronized to various hardware components so that processing can be distributed throughout the switch chassis. When distributed CEF mode is enabled, line cards maintain identical copies of the FIB and adjacency tables. The line cards perform the express forwarding between port adapters, relieving the RP of involvement in the switching operation, thus also enhancing system performance. Distributed CEF uses an interprocess communication (IPC) mechanism to ensure synchronization of FIB tables and adjacency tables on the RP and line cards. Figure 30-10 shows the relationship between the RP and line cards when distributed CEF is used.

Figure 30-9 Centralized Forwarding Architecture

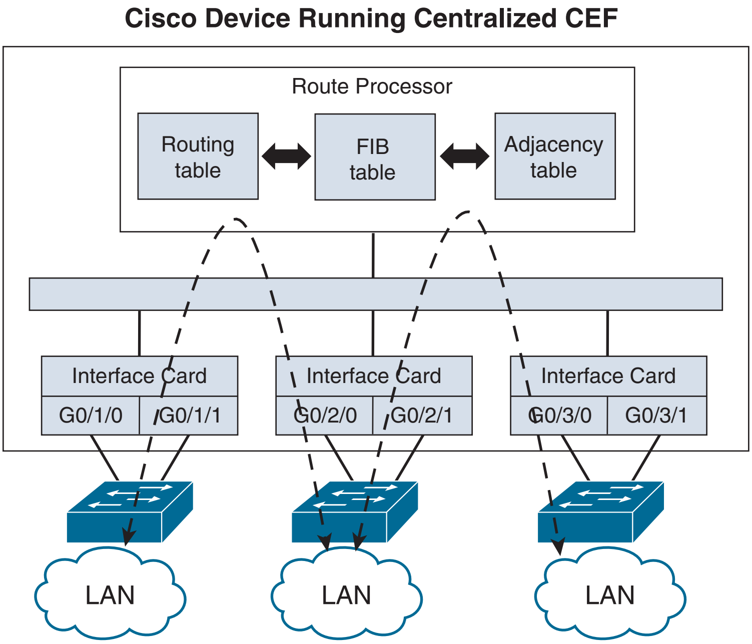
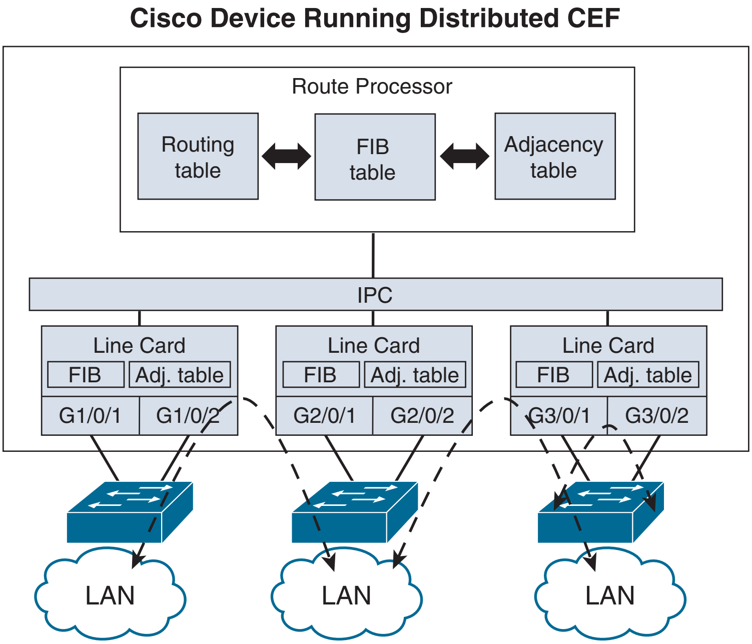

Cisco Device Running Centralized CEF
Route Processor
Routing table
FIB table
Adjacency table
Interface Card
G0/1/0
G0/1/1
Interface Card
G0/2/0
G0/2/1
Interface Card
G0/3/0
G0/3/1
LAN
LAN
LAN

Figure 30-10 Distributed Forwarding Architecture

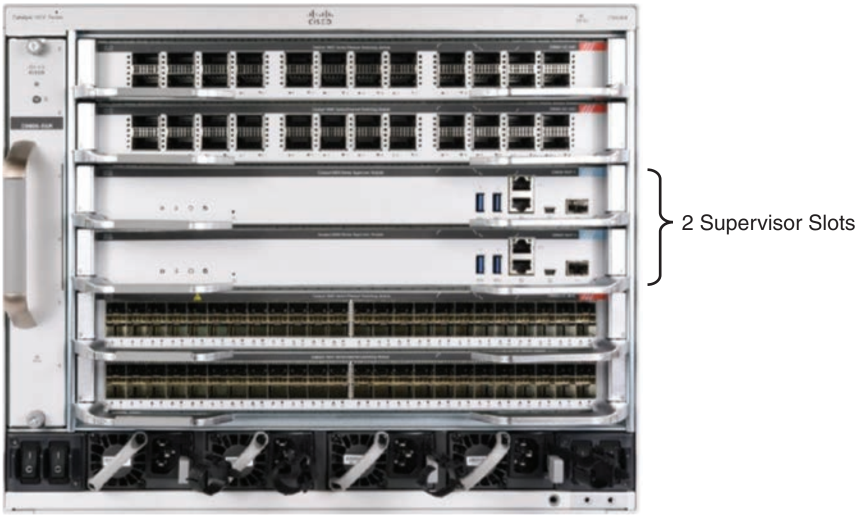

Cisco Device Running Distributed CEF
Route Processor
Routing table
FIB table
Adjacency table
IPC
Line Card
FIB
Adj. table
G1/0/1
G1/0/2
Line Card
FIB
Adj. table
G2/0/1
G2/0/2
Line Card
FIB
Adj. table
G3/0/1
G3/0/2
LAN
LAN
LAN

Hardware Redundancy Mechanisms

The Cisco Supervisor Engine module is the heart of the Cisco modular switch platform. The supervisor provides centralized forwarding information and processing. All software processes of a modular switch are run on a supervisor.

Platforms such as the Catalyst 4500, 6500, 6800, 9400, and 9600 Series switches can accept two supervisor modules that are installed in a single chassis; this setup prevents a single-point-of-failure situation. The first supervisor module to successfully boot becomes the active supervisor for the chassis. The other supervisor remains in a standby role, waiting for the active supervisor to fail. Figure 30-11 shows two supervisor modules installed in a Cisco Catalyst 9600 Series switch.

Figure 30-11 Cisco Catalyst 9600 Series Switch with Two Supervisors Installed



All switching functions are provided by the active supervisor. The standby supervisor, however, can boot up and initialize only to a certain level. When the active module fails, the standby module can proceed to initialize any remaining functions and take over the active role.

Redundant supervisor modules can be configured in several modes. The redundancy mode affects how the two supervisors handshake and synchronize information. Also, the mode limits the state of readiness for the standby supervisor. The more ready the standby module is allowed to become, the less initialization and failover time are required.

The following redundancy modes are available on modular Catalyst switches:

- **Route Processor Redundancy (RPR):** In this mode, the redundant supervisor is only partially booted and initialized. When the active module fails, the standby module must reload every other module in the switch and then initialize all the supervisor functions. Failover time is 2 to 4 minutes.
- **RPR+:** In this mode, the redundant supervisor is booted, allowing the supervisor and route engine to initialize. No Layer 2 or Layer 3 functions are started. When the active module fails,

the standby module finishes initializing without reloading other switch modules. This allows switch ports to retain their state. Failover time is 30 to 60 seconds.

- **Stateful Switchover (SSO):** In this mode, the redundant supervisor is fully booted and initialized. The startup and running configuration contents are synchronized between the supervisor modules. Layer 2 information is maintained on both supervisors so that hardware switching can continue during failover. The state of the switch interfaces is also maintained on both supervisors so that links do not flap during a failover. Failover time is 2 to 4 seconds.

Cisco Nonstop Forwarding

You can enable another redundancy feature along with SSO. Cisco Nonstop Forwarding (NSF) is an interactive method that focuses on quickly rebuilding the RIB table after a supervisor switchover. The RIB is used to generate the FIB table for CEF, which is downloaded to any switch module that can perform CEF.

Instead of waiting on any configured Layer 3 routing protocols to converge and rebuild the FIB, a router can use NSF to get assistance from other NSF-aware neighbors. The neighbors then can provide routing information to the standby supervisor, allowing the routing tables to be assembled quickly. In a nutshell, the Cisco NSF functions must be built into the routing protocols on both the router that will need assistance and the router that will provide assistance.

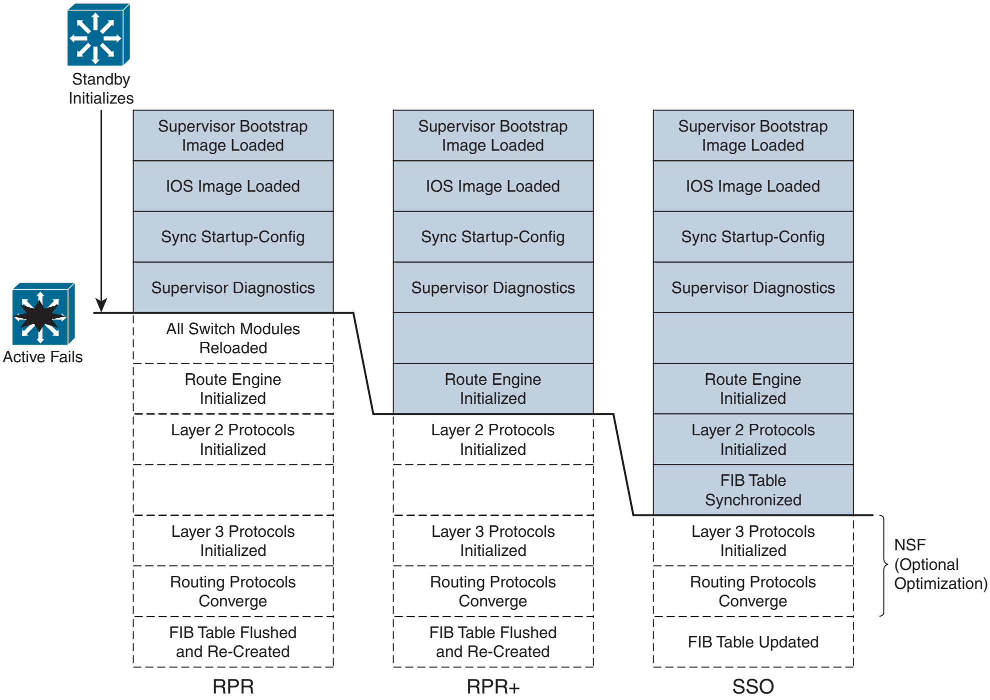
The stateful information is continuously synchronized from the active supervisor module to the standby supervisor module. This synchronization process uses a checkpoint facility between neighbors to ensure that the link state and Layer 2 protocol details are mirrored on the standby route processor (RP). Switching over to the standby RP takes 150 ms or less. There are less than 200 ms of traffic interruption. On Catalyst 9000 Series switches, the failover time between supervisors within the same chassis can be less than 5 ms.

SSO with NSF minimizes the time a network is unavailable to users following a switchover while continuing the nonstop forwarding of IP packets. The user session information is maintained during a switchover, and line cards continue to forward network traffic with no loss of sessions.

NSF is supported by Border Gateway Protocol (BGP), Enhanced Interior Gateway Routing Protocol (EIGRP), Open Shortest Path First (OSPF), and Intermediate System-to-Intermediate System (IS-IS) routing protocols.

Figure 30-12 shows how the supervisor redundancy modes compare with respect to the functions they perform. The shaded functions are performed as the standby supervisor initializes and then waits for the active supervisor to fail. When a failure is detected, the remaining functions must be performed in sequence before the standby supervisor can become fully active. Notice that the redundancy modes get progressively more initialized and ready to become active and that NSF focuses on Layer 3 routing protocol synchronization.

Figure 30-12 Standby Supervisor Readiness as a Function of Redundancy Mode


Standby Initializes
Active Fails
Supervisor Bootstrap Image Loaded
IOS Image Loaded
Sync Startup-Config
Supervisor Diagnostics
All Switch Modules Reloaded
Route Engine Initialized
Layer 2 Protocols Initialized
Layer 3 Protocols Initialized
Routing Protocols Converge
FIB Table Flushed and Re-Created
RPR
Supervisor Bootstrap Image Loaded
IOS Image Loaded
Sync Startup-Config
Supervisor Diagnostics
Route Engine Initialized
Layer 2 Protocols Initialized
Layer 3 Protocols Initialized
Routing Protocols Converge
FIB Table Flushed and Re-Created
RPR+
Supervisor Bootstrap Image Loaded
IOS Image Loaded
Sync Startup-Config
Supervisor Diagnostics
Route Engine Initialized
Layer 2 Protocols Initialized
FIB Table Synchronized
Layer 3 Protocols Initialized
Routing Protocols Converge
FIB Table Updated
SSO
NSF (Optional Optimization)

SDM Templates

Access layer switches were not built to be used in routing OSPFv3 or BGP, even though they could be used for that implementation as well. By default, the resources of these switches are allocated to a more common set of tasks. If you want to use an access layer switch for something other than the default common set of tasks, you can use its option for reallocation of resources.

You can use SDM templates to configure system resources (CAM and TCAM) in a switch to optimize support for specific features, depending on how the switch is used in the network. You can select a template to provide maximum system usage for some functions; for example, you can use the default template to balance resources and use access templates to obtain maximum ACL usage. To allocate hardware resources for different usages, the switch SDM templates prioritize system resources to optimize support for certain features.

You can verify the SDM template that is in use with the show sdm prefer command. Available SDM templates depend on the device type and Cisco IOS XE Software version that is used. Table 30-2 summarizes the SDM templates available on different Cisco IOS XE Catalyst switches.

Table 30-2 SDM Templates by Switch Model

Switch Model	SDM Template Name	Description
3650 3850 9200	Advanced	The advanced template maximizes system resources for features like NetFlow, multicast groups, security ACEs, QoS ACEs, and so on.
	VLAN	The VLAN template disables routing and supports the maximum number of unicast MAC addresses. It is typically selected for a Layer 2 device.
9300	Access	The access template maximizes resources for access layer functionality (VLANs, MAC addresses).
	NAT	The NAT template explicitly reserves more memory for PBR and NAT functionality.
9400	Access	The access template maximizes resources for access layer functionality (VLANs, MAC addresses).
	Core	The core template allows for the storage of a very large number of IPv4 and IPv6 unicast and multicast routes.
	SDA	The SDA template is similar to the core template, except that it also explicitly reserves more memory for hosts.
	NAT	The NAT template is similar to the core template, except that it also explicitly reserves more memory for hosts, as well as PBR and NAT functionality.
9500 9600	Distribution	The distribution template maximizes the storage of MAC addresses in the MAC address table, and it maximizes the number of flows for Flexible NetFlow.
	Core	The core template allows for the storage of a very large number of IPv4 and IPv6 unicast and multicast routes.
	SDA	The SDA template is similar to the core template, except that it also explicitly reserves more memory for egress security ACLs and LISP functionality.
	NAT	The NAT template is similar to the core template, except that it also explicitly reserves more memory for PBR and NAT functionality.

The most common reason for changing the SDM template on older IOS-based Catalyst switches is to enable IPv6 routing. Using the dual-stack template results in less TCAM capacity for other resources.

Another common reason for changing the SDM template is that the switch is low on resources. For example, a switch might have so many access lists that you need to change to the access SDM template. In this case, it is important to first investigate whether you can optimize the performance so that you do not need to change the SDM template. It might be that the ACLs that you are using are set up inefficiently—for example, with redundant entries, most common entries at the end of the list, unnecessary entries, and so on. Changing the SDM template reallocates internal resources from one function to another, correcting one issue (ACLs), while perhaps inadvertently causing a new separate issue elsewhere in the switch (IPv4 routing).

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	**Module, Chapter, or Link**
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	1
System Management Configuration Guide, Cisco IOS XE	https://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst9600/software/release/16-12/configuration_guide/sys_mgmt/b_1612_sys_mgmt_9600_cg.html
High Availability Configuration Guide, Cisco IOS XE	https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ha/configuration/xe-17/ha-xe-17-book.html

Day 29

LAN Connectivity

ENCOR 350-401 Exam Topics

Infrastructure

- Layer 2
  - Troubleshoot static and dynamic 802.1q trunking protocols

Key Topics

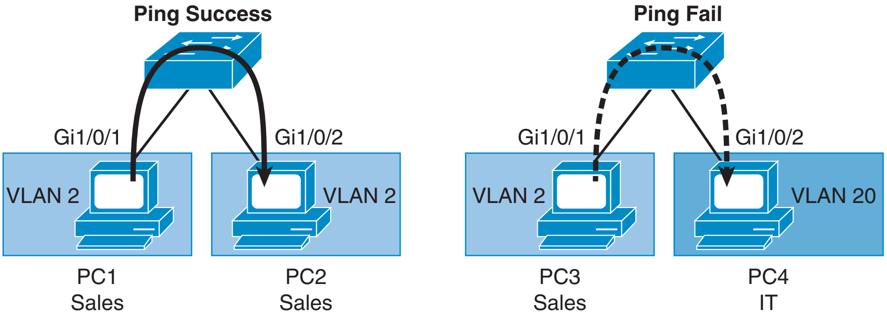
Today we review concepts related to configuring, verifying, and troubleshooting VLANs, 802.1Q trunking, Dynamic Trunking Protocol (DTP), VLAN Trunking Protocol (VTP), and inter-VLAN routing using a router and a Layer 3 switch.

VLAN Overview

A VLAN is a logical broadcast domain that can span multiple physical LAN segments. Within a switched internetwork, VLANs provide segmentation and organizational flexibility. You can design a VLAN structure that lets you group stations that are segmented logically by functions, project teams, and applications without regard to the physical location of the users. Ports in the same VLAN share broadcasts. Ports in different VLANs do not share broadcasts. Containing broadcasts within a VLAN improves the overall performance of the network.

Each VLAN that you configure on a switch implements address learning, forwarding, and filtering decisions and loop-avoidance mechanisms, just as though the VLAN were a separate physical bridge. A Cisco Catalyst switch implements VLANs by restricting traffic forwarding to destination ports that are in the same VLAN as the originating ports. When a frame arrives on a switch port, the switch must retransmit the frame only to the ports that belong to the same VLAN. A VLAN that is operating on a switch limits transmission of unicast, multicast, and broadcast traffic, as shown in Figure 29-1, where traffic is forwarded between devices within the same VLAN, in this case VLAN 2, while traffic is not forwarded between devices in different VLANs.

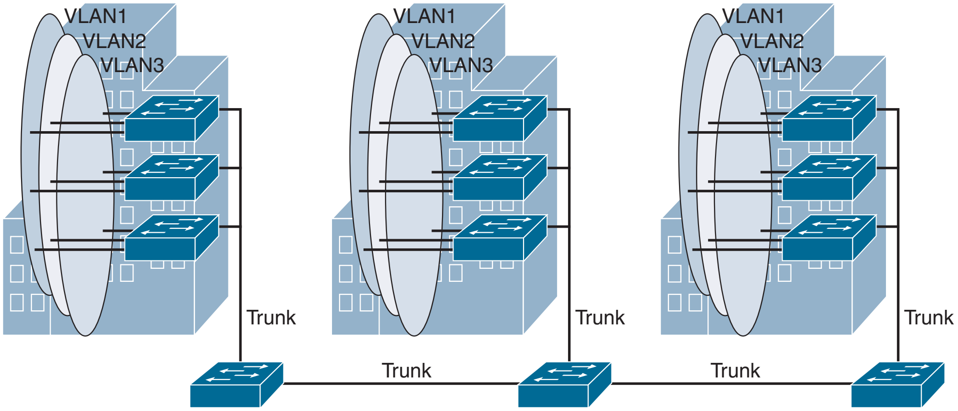
Figure 29-1 VLAN Traffic Patterns



A VLAN can exist on a single switch or span multiple switches. VLANs can include stations in single- or multiple-building infrastructures. The process of forwarding network traffic from one VLAN to another VLAN using a router or Layer 3 switch is called *inter-VLAN routing*. In a campus design, a network administrator can design a campus network with one of two models: end-to-end VLANs or local VLANs.

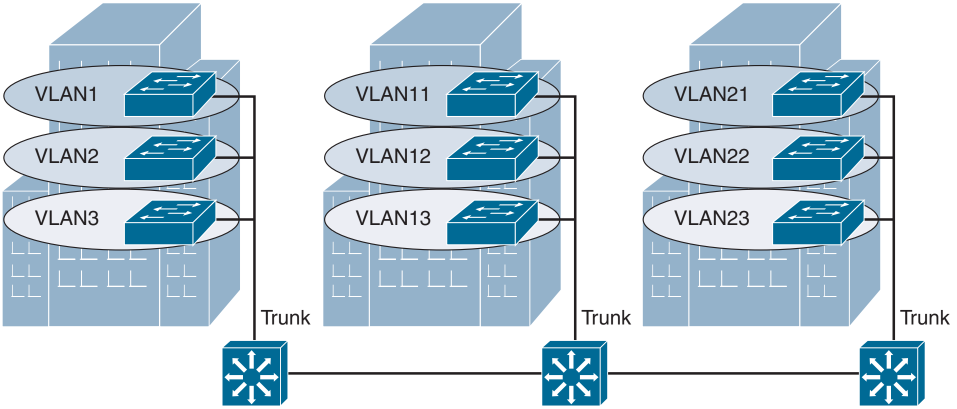
The term *end-to-end VLAN* refers to a single VLAN that is associated with switch ports widely dispersed throughout an enterprise network on multiple switches. A Layer 2 switched campus network carries traffic for this VLAN throughout the network, as shown in Figure 29-2, where VLANs 1, 2, and 3 are spread across all three switches.

Figure 29-2 End-to-End VLANs



The typical campus enterprise architecture is usually based on the local VLAN model instead. In a local VLAN model, all users of a set of geographically common switches are grouped into a single VLAN, regardless of the organizational function of those users. Local VLANs are generally confined to a wiring closet, as shown in Figure 29-3. In the local VLAN model, Layer 2 switching is implemented at the access level, and routing is implemented at the distribution and core levels, as discussed on Day 31, "Enterprise Network Architecture," to enable users to maintain access to the resources they need. An alternative design is to extend routing to the access layer, with routed links between the access switches and distribution switches. In Figure 29-3, notice the use of trunk links between switches and buildings. These are special links that can carry traffic for all VLANs. Trunking is explained in greater detail later today.

Figure 29-3 Local VLANs



Creating a VLAN

To create a VLAN, use the **vlan** global configuration command and enter the VLAN configuration mode. Use the **no** form of this command to delete the VLAN. Example 29-1 shows how to add VLAN 2 to the VLAN database and how to name it Sales. VLAN 20 is also created, and it is named IT.

Example 29-1 Creating a VLAN

```
Switch# configure terminal
Switch(config)# vlan 2
Switch(config-vlan)# name Sales
Switch(config-vlan)# vlan 20
Switch(config-vlan)# name IT
```

To add a VLAN to the VLAN database, assign a number and name to the VLAN. VLAN 1 is the factory default VLAN. Normal-range VLANs are identified with a number between 1 and 1001. The VLAN numbers 1002 through 1005 are reserved. VIDs 1 and 1002 to 1005 are automatically created, and you cannot remove them. The extended VLAN range is from 1006 to 4094. The configurations for VLANs 1 to 1005 are written to the vlan.dat file (VLAN database). You can display the VLANs by entering the **show vlan** command in privileged EXEC mode. The vlan.dat file is stored in flash memory.

Access Ports

When you connect an end system to a switch port, you should associate it with a VLAN in accordance with the network design. This procedure allows frames from that end system to be forwarded to other interfaces that also function on that VLAN. To associate a device with a VLAN, assign the switch port to which the device connects to a single-data VLAN. The switch port, therefore, becomes an access port. By default, all ports are members of VLAN 1. In Example 29-2, the GigabitEthernet 1/0/5 interface is assigned to VLAN 2, and the GigabitEthernet 1/0/15 interface is assigned to VLAN 20.

Example 29-2 Assigning a Port to a VLAN

```
Switch# configure terminal
Switch(config)# interface GigabitEthernet 1/0/5
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 2
Switch(config-if)# interface GigabitEthernet 1/0/15
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 20
```

After creating a VLAN, you can manually assign a port or many ports to this VLAN. An access port can belong to only one VLAN at a time.

Use the **show vlan** or **show vlan brief** command to display information about all configured VLANs, or use either the **show vlan id** *vlan_number* or **show vlan name** *vlan-name* command to display information about specific VLANs in the VLAN database, as shown in Example 29-3.

Example 29-3 Using *show vlan* Commands

```
Switch# show vlan

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Gi1/0/1, Gi1/0/2, Gi1/0/3
                                                Gi1/0/4, Gi1/0/6, Gi1/0/7
                                                Gi1/0/8, Gi1/0/9, Gi1/0/10
                                                Gi1/0/11, Gi1/0/12, Gi1/0/13
                                                Gi1/0/14, Gi1/0/16, Gi1/0/17
                                                Gi1/0/18, Gi1/0/19, Gi1/0/20
                                              Gi1/0/21, Gi1/0/22, Gi1/0/23
                                              Gi1/0/24
2    Sales                            active    Gi1/0/5
20   IT                               active    Gi1/0/15
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------  ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        0      0
2    enet  100002     1500  -      -      -        -    -        0      0
20   enet  100020     1500  -      -      -        -    -        0      0
1002 fddi  101002     1500  -      -      -        -    -        0      0
1003 tr    101003     1500  -      -      -        -    -        0      0
1004 fdnet 101004     1500  -      -      -        ieee -        0      0
1005 trnet 101005     1500  -      -      -        ibm  -        0      0
```

```
Primary Secondary Type              Ports
------- --------- ----------------- ------------------------------------------
```

```
Switch# show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- ------------------------------
1    default                          active    Gi1/0/1, Gi1/0/2, Gi1/0/3
                                                Gi1/0/4, Gi1/0/6, Gi1/0/7
                                                Gi1/0/8, Gi1/0/9, Gi1/0/10
                                                Gi1/0/11, Gi1/0/12, Gi1/0/13
                                                Gi1/0/14, Gi1/0/16, Gi1/0/17
                                                Gi1/0/18, Gi1/0/19, Gi1/0/20
                                              Gi1/0/21, Gi1/0/22, Gi1/0/23
                                              Gi1/0/24
2    Sales                            active    Gi1/0/5
20   IT                               active    Gi1/0/15
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup
```

```
Switch# show vlan id 2

VLAN Name                 Status    Ports
---- -------------------- -------   ---------------------
2    Sales                 active   Gi1/0/5

VLAN Type SAID    MTU    Parent RingNo BridgeNo Stp BrdgMode  Trans1 Trans2
---- ---- ------- ----- ------ ------ -------- --- --------- ------ ------
2    enet 100002  1500   -       -      -         -    -         0      0

<... output omitted ...>
```

```
Switch# show vlan name IT

VLAN Name                 Status    Ports
---- -------------------- -------   ---------------------
20   IT                   active    Gi1/0/15

VLAN Type SAID    MTU   Parent RingNo BridgeNo Stp BrdgMode Trans1 Trans2
---- ---- ------- ----- ------ ------ -------- --- --------- ------ ------
2    enet 100002  1500   -       -      -         -    -         0      0

<... output omitted ...>
```

Use the **show interfaces switchport** command to display switch port status and characteristics. The output in Example 29-4 shows information about the GigabitEthernet 1/0/5 interface, where VLAN 2 (Sales) is assigned and the interface is configured as an access port.

Example 29-4 Using the *show interfaces switchport* Command

```
Switch# show interfaces GigabitEthernet 1/0/5 switchport
Name: Gi1/0/5
Switchport: Enabled
Administrative Mode: static access
Operational Mode: static access
Administrative Trunking Encapsulation: dot1q
Negotiation of Trunking: On
Access Mode VLAN: 2 (Sales)
Trunking Native Mode VLAN: 1 (default)
Administrative Native VLAN tagging: enabled
Voice VLAN: none
Administrative private-vlan host-association: none
Administrative private-vlan mapping: none
Administrative private-vlan trunk native VLAN: none
Administrative private-vlan trunk Native VLAN tagging: enabled
Administrative private-vlan trunk encapsulation: dot1q
Administrative private-vlan trunk normal VLANs: none
Administrative private-vlan trunk associations: none
Administrative private-vlan trunk mappings: none
Operational private-vlan: none
Trunking VLANs Enabled: ALL
Pruning VLANs Enabled: 2-1001
Capture Mode Disabled
Capture VLANs Allowed: ALL

Protected: false
Unknown unicast blocked: disabled
Unknown multicast blocked: disabled
Appliance trust: none
```

802.1Q Trunk Ports

A port normally carries only the traffic for a single VLAN. For a VLAN to span multiple switches, a trunk is required to connect the switches. A trunk can carry traffic for multiple VLANs.

A trunk is a point-to-point link between one or more Ethernet switch interfaces and another networking device, such as a router or a switch. An Ethernet trunk carries the traffic of multiple VLANs over a single link and allows you to extend the VLANs across an entire network. A trunk does not belong to a specific VLAN; rather, it is a conduit for VLANs between switches and routers.

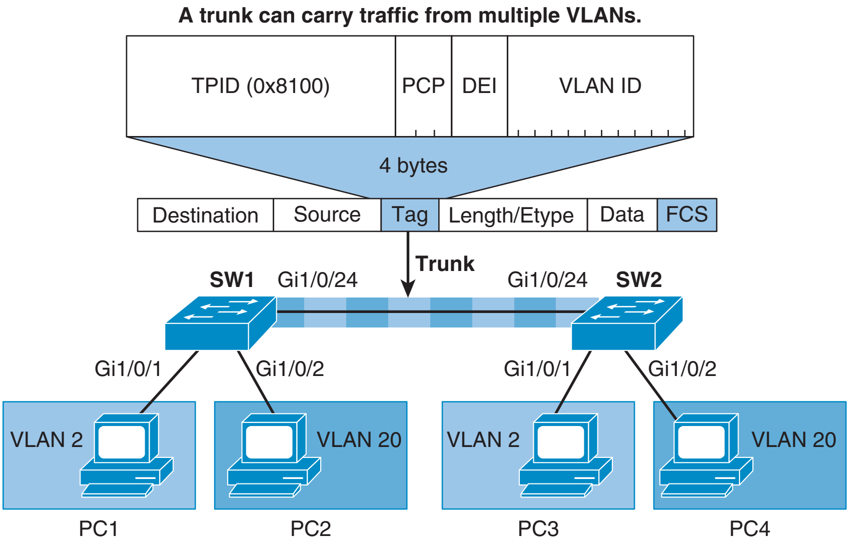
A special protocol is used to carry multiple VLANs over a single link between two devices. There are two trunking technologies: ISL and IEEE 802.1Q. ISL is a Cisco-proprietary implementation that is no longer widely used. The 802.1Q technology is the IEEE standard VLAN trunking protocol. This protocol inserts a 4-byte tag into the original Ethernet header and then recalculates and updates the FCS in the original frame and transmits the frame over the trunk link. A trunk could also be used between a network device and server or another device that is equipped with an appropriate 802.1Q-capable NIC.

Ethernet trunk interfaces support various trunking modes. You can configure an interface as trunking or nontrunking, and you can have an interface negotiate trunking with the neighboring interface.

By default, all configured VLANs are carried over a trunk interface on a Cisco Catalyst switch. On an 802.1Q trunk port, there is one native VLAN, which is untagged (by default, VLAN 1). Each of the other VLANs is tagged with a VID.

When Ethernet frames are placed on a trunk, they need additional information about the VLANs they belong to. This task is accomplished by using the 802.1Q encapsulation header. It is the responsibility of the Ethernet switch to look at the 4-byte tag field and determine where to deliver the frame. Figure 29-4 illustrates the tagging process that occurs on the Ethernet frame as it is placed on the 802.1Q trunk.

Figure 29-4 802.1Q Tagging Process


A trunk can carry traffic from multiple VLANs.
TPID (0x8100)
PCP
DEI
VLAN ID
4 bytes
Destination
Source
Tag
Length/Etype
Data
FCS
Trunk
SW1
Gi1/0/24
Gi1/0/24
SW2
Gi1/0/1
Gi1/0/2
Gi1/0/1
Gi1/0/2
VLAN 2
VLAN 20
VLAN 2
VLAN 20
PC1
PC2
PC3
PC4

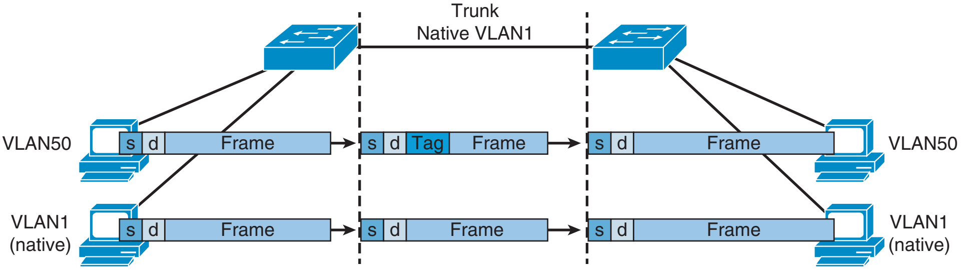
According to the IEEE 802.1Q-2018 revision of the 802.1Q standard, the tag has these four components:

- **Tag Protocol Identifier (TPID; 16 bits):** Uses EtherType 0x8100 to indicate that this frame is an 802.1Q frame.
- **Priority Code Point (PCP; 3 bits):** Carries the class of service (CoS) priority information for Layer 2 quality of service (QoS). Different PCP values can be used to prioritize different classes of traffic.
- **Drop Eligible Indicator (DEI; 1 bit):** Formerly called CFI. May be used separately or in conjunction with PCP to indicate frames eligible to be dropped in the presence of congestion.
- **VLAN Identifier (VID; 12 bits):** VLAN association of the frame. The hexadecimal values 0x000 and 0xFFF are reserved. All other values may be used as VLAN identifiers, allowing up to 4094 VLANs.

Native VLAN

The IEEE 802.1Q protocol allows operation between equipment from different vendors. All frames, except native VLAN, are equipped with a tag when traversing the link, as shown in Figure 29-5.

Figure 29-5 Native VLAN in 802.1Q



A frequent configuration error is to have different native VLANs. The native VLANs configured on each end of an 802.1Q trunk must be the same. If one end is configured for native VLAN 1 and the other for native VLAN 2, a frame that is sent in VLAN 1 on one side will be received on VLAN 2 on the other as VLAN 1 and VLAN 2 have been segmented and merged. This configuration will lead to connectivity issues in the network. If there is a native VLAN mismatch on either side of an 802.1Q link, Layer 2 loops may occur because VLAN 1 STP BPDUs are sent to the IEEE STP MAC address (0180.c200.0000) untagged.

Cisco switches use Cisco Discovery Protocol (CDP) to warn about native VLAN mismatches. By default, the native VLAN is VLAN 1. For security purposes, the native VLAN on a trunk should be set to a specific VID that is not used for normal operations elsewhere on the network.

Allowed VLANs

By default, a switch transports all active VLANs (1 to 4094) over a trunk link. An active VLAN is one that has been defined on the switch and that has ports assigned to carry it. There might be times when the trunk link should not carry all VLANs. For example, say that broadcasts are forwarded to every switch port on a VLAN—including a trunk link because it, too, is a member of the VLAN. If the VLAN does not extend past the far end of the trunk link, propagating broadcasts across the trunk makes no sense and only wastes trunk bandwidth.

802.1Q Trunk Configuration

Example 29-5 shows GigabitEthernet 1/0/24 being configured as a trunk port using the **switchport mode trunk** interface-level command.

Example 29-5 Configuring an 802.1Q Trunk Port

```
Switch# configure terminal
Switch(config)# interface GigabitEthernet 1/0/24
Switch(config-if) switchport mode trunk
Switch(config-if) switchport trunk native vlan 900
Switch(config-if) switchport trunk allowed vlan 1,2,20,900
```

In Example 29-5, the interface is configured with the **switchport trunk native vlan** command to use VLAN 900 as the native VLAN.

You can tailor the list of allowed VLANs on the trunk by using the **switchport trunk allowed vlan** command with one of the following keywords:

- *vlan-list*: Specifies an explicit list of VLAN numbers, separated by commas or dashes.
- **all**: Indicates that all active VLANs (1 to 4094) will be allowed.
- **add** *vlan-list*: Specifies a list of VLAN numbers to add to the already configured list.
- **except** *vlan-list*: Indicates that all VLANs (1 to 4094) will be allowed, except for the VLAN numbers listed.
- **remove** *vlan-list*: Specifies a list of VLAN numbers that will be removed from the already configured list.

In Example 29-5, only VLANs 1, 2, 20, and 900 are permitted across the GigabitEthernet 1/0/24 trunk link.

NOTE: On some Catalyst switch models, you might need to manually configure the 802.1Q trunk encapsulation protocol before enabling trunking. Use the **switchport trunk encapsulation dot1q** command to achieve this.

802.1Q Trunk Verification

To view the trunking status on a switch port, use the **show interfaces trunk** and **show interfaces switchport** commands, as demonstrated in Example 29-6:

Example 29-6 Verifying 802.1Q Trunking

```
Switch# show interfaces trunk
Port      Mode          Encapsulation  Status        Native vlan
Gi1/0/24  on            802.1q         trunking      900

Port      Vlans allowed on trunk
Gi1/0/24  1,2,20,900

Port      Vlans allowed and active in management domain
Gi1/0/24  1,2,20,900

Port      Vlans in spanning tree forwarding state and not pruned
Gi1/0/24  1,2,20,900
```

```
Switch# show interfaces GigabitEthernet 1/0/24 switchport
Name: Gi1/0/24
Switchport: Enabled
Administrative Mode: trunk
Operational Mode: trunk
Administrative Trunking Encapsulation: dot1q
Operational Trunking Encapsulation: dot1q
Negotiation of Trunking: On
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 900 (Native)
Administrative Native VLAN tagging: enabled
Voice VLAN: none
Administrative private-vlan host-association: none
Administrative private-vlan mapping: none
Administrative private-vlan trunk native VLAN: none
Administrative private-vlan trunk Native VLAN tagging: enabled
Administrative private-vlan trunk encapsulation: dot1q
Administrative private-vlan trunk normal VLANs: none
Administrative private-vlan trunk associations: none
Administrative private-vlan trunk mappings: none
Operational private-vlan: none
Trunking VLANs Enabled: 1,2,20,900
Pruning VLANs Enabled: 2-1001
Capture Mode Disabled
Capture VLANs Allowed: ALL

Protected: false
Unknown unicast blocked: disabled
Unknown multicast blocked: disabled
Appliance trust: none
```

The **show interfaces trunk** command lists all the interfaces on the switch that are configured and operating as trunks. The output also confirms the trunk encapsulation protocol (802.1Q), the native VLAN, and which VLANs are allowed across the link. The **show interfaces switchport** command provides similar information.

Another command that is useful for verifying both access and trunk port Layer 1 and Layer 2 status is the **show interfaces status** command, as show in Example 29-7.

Example 29-7 Verifying the Switch Port Status

```
Switch# show interfaces trunk

Port      Name               Status       Vlan       Duplex  Speed  Type
Gig1/0/1                     notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/2                     notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/3                     notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/4                     notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/5                     connected    2          a-full  a-1000 10/100/1000BaseTX
Gig1/0/6                     notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/7                     notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/8                     notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/9                     notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/10                    notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/11                    notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/12                    notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/13                    notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/14                    notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/15                    connected    20         a-full  a-1000 10/100/1000BaseTX
Gig1/0/16                    notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/17                    notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/18                    notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/19                    notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/20                    notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/21                    notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/22                    notconnect   1          auto    auto   10/100/1000BaseTX
Gig1/0/23                    disabled     999        auto    auto   10/100/1000BaseTX
Gig1/0/24                    connected    trunk      a-full  a-1000 10/100/1000BaseTX
```

In the output in Example 29-7, interface GigabitEthernet 1/0/5 is configured for VLAN 2, GigabitEthernet 1/0/15 is configured for VLAN 20, and GigabitEthernet 1/0/24 is configured as a trunk. The Status column refers to the Layer 1 state of the interface. Notice in the output that interface GigabitEthernet 1/0/23 is disabled. This is displayed when an interface is administratively shut down.

Dynamic Trunking Protocol

Cisco switch ports can run Dynamic Trunking Protocol (DTP), which can automatically negotiate a trunk link. This Cisco-proprietary protocol can determine an operational trunking mode and protocol on a switch port when it is connected to another device that is also capable of dynamic trunk negotiation.

There are three modes you can use with the **switchport mode** command when configuring a switch port to trunk:

- **Trunk:** The **trunk** setting places a port in permanent trunking mode. DTP is still operational, so if the far-end switch port is configured to trunk, dynamic desirable, or dynamic auto mode, trunking will be negotiated successfully. The trunk mode is usually used to establish an unconditional trunk. Therefore, the corresponding switch port at the other end of the trunk should be configured similarly. In this way, both switches always expect the trunk link to be operational without any negotiation. Use the **switchport mode trunk** command to achieve this.
- **Dynamic desirable:** With this mode, the port actively attempts to convert the link into trunking mode. In other words, it asks the far-end switch to bring up a trunk. If the far-end switch port is configured to trunk, dynamic desirable, or dynamic auto mode, trunking is negotiated successfully. Use the **switchport mode dynamic desirable** command to achieve this.
- **Dynamic auto:** With this mode, the port can be converted into a trunk link—but only if the far-end switch actively requests it. Therefore, if the far-end switch port is configured to trunk or dynamic desirable mode, trunking is negotiated. Because of the passive negotiation behavior, the link never becomes a trunk if both ends of the link are set to dynamic auto mode. Use the **switchport mode dynamic auto** command to achieve this.

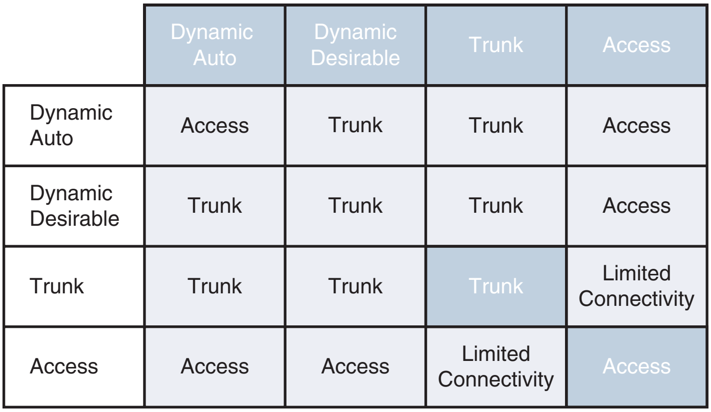
The default DTP mode depends on the Cisco IOS Software version and on the platform. To determine the current DTP mode of an interface, issue the **show interfaces switchport** command, as illustrated in Example 29-8.

Example 29-8 Verifying DTP Status

```
Switch# show interfaces GigabitEthernet 1/0/10
Name: Gi1/0/10
Switchport: Enabled
Administrative Mode: dynamic auto
Operational Mode: down
Administrative Trunking Encapsulation: dot1q
Negotiation of Trunking: On
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 1 (default)
Administrative Native VLAN tagging: enabled
<... output omitted ...>
```

In the output in Example 29-8, the GigabitEthernet 1/0/10 interface is currently configured in dynamic auto mode, but the operational mode is down because the interface is not connected. If it were connected to another switch running DTP, its operational state would change to either static access or trunking once negotiation was successfully completed. Figure 29-6 shows the combination of DTP modes between the two links. A combination of DTP modes can produce either an access port or a trunk port.

Figure 29-6 DTP Combinations



	Dynamic Auto	Dynamic Desirable	Trunk	Access
Dynamic Auto	Access	Trunk	Trunk	Access
Dynamic Desirable	Trunk	Trunk	Trunk	Access
Trunk	Trunk	Trunk	Trunk	Limited Connectivity
Access	Access	Access	Limited Connectivity	Access

Notice that Figure 29-6 also includes access as a DTP mode. Using the **switchport mode access** command puts the interface into a permanent nontrunking mode and negotiates to convert the link into a nontrunking link.

In all these modes, DTP frames are sent out every 30 seconds to keep neighboring switch ports informed of a link's mode. On critical trunk links in a network, manually configuring the trunking mode on both ends is best so that the link can never be negotiated to any other state.

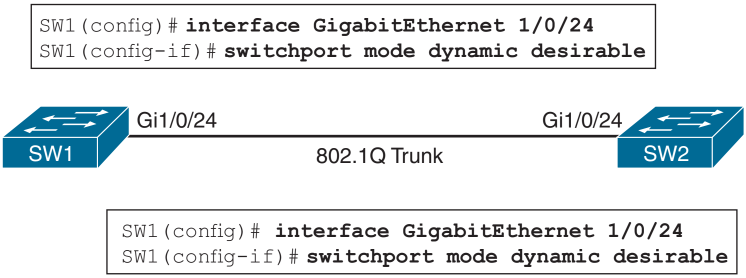
As a best practice, you should configure both ends of a trunk link as a fixed trunk (**switchport mode trunk**) or as an access link (**switchport mode access**) to remove any uncertainty about the link operation. In the case of a trunk, you can disable DTP completely so that the negotiation frames are not exchanged at all. To do this, add the **switchport nonegotiate** command to the interface configuration. Be aware that after DTP frames are disabled, no future negotiation is possible until this configuration is reversed.

DTP Configuration Example

Figure 29-7 illustrates a topology in which SW1 and SW2 use a combination of DTP modes to establish an 802.1Q trunk.

In this example, SW1 is configured to actively negotiate a trunk with SW2. SW2 is configured to passively negotiate a trunk with SW1. Example 29-9 shows confirmation that an 802.1Q trunk is successfully negotiated.

Figure 29-7 DTP Configuration Example Topology



```
SW1(config)# interface GigabitEthernet 1/0/24
SW1(config-if)# switchport mode dynamic desirable
```

```
SW1(config)#  interface GigabitEthernet 1/0/24
SW1(config-if)# switchport mode dynamic desirable
```

Example 29-9 Verifying Trunk Status Using DTP

```
SW1# show interfaces trunk
Port         Mode              Encapsulation  Status        Native vlan
Gi1/0/24     desirable         802.1q         trunking      1

Port         Vlans allowed on trunk
Gi1/0/24     1-4094

Port         Vlans allowed and active in management domain
Gi1/0/24     1-4094

Port         Vlans in spanning tree forwarding state and not pruned
Gi1/0/24     1-4094
```

```
SW2# show interfaces trunk
Port         Mode              Encapsulation  Status        Native vlan
Gi1/0/24     auto              802.1q         trunking      1

Port         Vlans allowed on trunk
Gi1/0/24     1-4094

Port         Vlans allowed and active in management domain
Gi1/0/24     1-4094

Port         Vlans in spanning tree forwarding state and not pruned
Gi1/0/24     1-4094
```

VLAN Trunking Protocol

VLAN Trunking Protocol (VTP) is a Layer 2 protocol that maintains VLAN configuration consistency by managing the additions, deletions, and name changes of VLANs across networks. VTP is organized into management domains, or areas with common VLAN requirements. A switch can belong to only one VTP domain, and it shares VLAN information with other switches in the domain. Switches in different VTP domains, however, do not share VTP information. Switches in a VTP domain advertise several attributes to their domain neighbors. Each advertisement contains information about the VTP management domain, VTP revision number, known VLANs, and specific VLAN parameters. When a VLAN is added to a switch in a management domain, other switches are notified of the new VLAN through VTP advertisements. In this way, all switches in a domain can prepare to receive traffic on their trunk ports using the new VLAN.

VTP Modes

To participate in a VTP management domain, each switch must be configured to operate in one of several modes. The VTP mode determines how the switch processes and advertises VTP information. You can use the following modes:

- **Server mode:** VTP servers have full control over VLAN creation and modification for their domains. All VTP information is advertised to other switches in the domain, and all received VTP information is synchronized with the other switches. By default, a switch is in VTP server mode. Note that each VTP domain must have at least one server so that VLANs can be created, modified, or deleted and so VLAN information can be propagated.

- **Client mode:** VTP clients do not allow the administrator to create, change, or delete any VLANs. Instead, they listen to VTP advertisements from other switches and modify their VLAN configurations accordingly. In effect, this is a passive listening mode. Received VTP information is forwarded out trunk links to neighboring switches in the domain, so the switch also acts as a VTP relay.

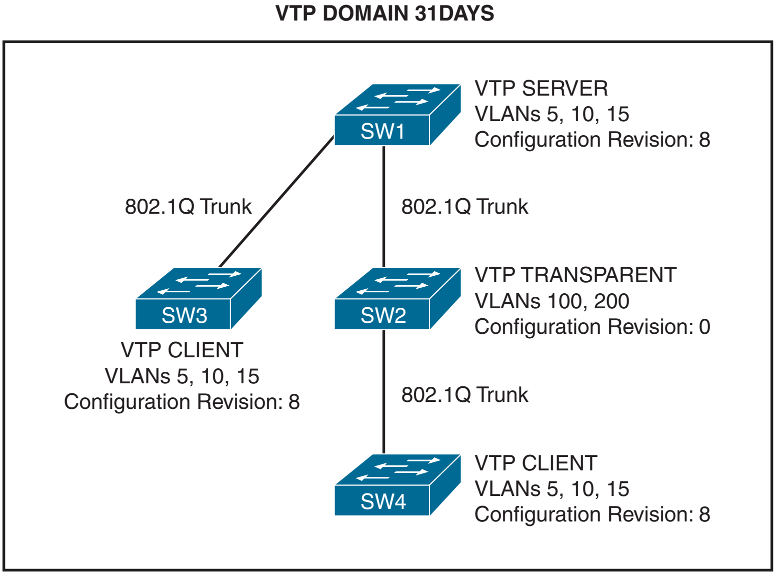
- **Transparent mode:** VTP transparent switches do not participate in VTP. While in transparent mode, a switch does not advertise its own VLAN configuration, and it does not synchronize its VLAN database with received advertisements.

- **Off mode:** Like transparent mode, switches in VTP off mode do not participate in VTP; however, VTP advertisements are not relayed at all. You can use VTP off mode to disable all VTP activity on or through a switch.

Figure 29-8 illustrates a simple network in which SW1 is the VTP server for the domain 31DAYS. SW3 and SW4 are configured as VTP clients, and SW2 is configured as VTP transparent. SW1, SW3, and SW4 have synchronized VLAN databases with VLANs 5, 10, and 15. SW2 has propagated VTP information to SW4, but its own database only contains VLANs 100 and 200.

VTP advertisements are flooded throughout the management domain. VTP summary advertisements are sent every 5 minutes or whenever there is a change in VLAN configuration. Advertisements are transmitted (untagged) over the native VLAN (VLAN 1 by default) using a multicast frame.

Figure 29-8 VTP Topology Example


VTP DOMAIN 31DAYS
SW1
VTP SERVER
VLANs 5, 10, 15
Configuration Revision: 8
802.1Q Trunk
802.1Q Trunk
SW3
VTP CLIENT
VLANs 5, 10, 15
Configuration Revision: 8
SW2
VTP TRANSPARENT
VLANs 100, 200
Configuration Revision: 0
802.1Q Trunk
SW4
VTP CLIENT
VLANs 5, 10, 15
Configuration Revision: 8

VTP Configuration Revision

One of the most critical components of VTP is the configuration revision number. Each time a VTP server modifies its VLAN information, the VTP server increments the configuration revision number by one. The server then sends out a VTP subset advertisement with the new configuration revision number. If the configuration revision number being advertised is higher than the number stored on the other switches in the VTP domain, the switches overwrite their VLAN configurations with the new information that is being advertised. The configuration revision number in VTP transparent mode is always 0.

A device that receives VTP advertisements must check various parameters before incorporating the received VLAN information. First, the management domain name and password in the advertisement must match the values that are configured on the local switch. Next, if the configuration revision number indicates that the message was created after the configuration currently in use, the switch incorporates the advertised VLAN information.

Returning to the example in Figure 29-8, notice that the current configuration revision number is 8. If a network administrator were to add a new VLAN to the VTP server (SW1), the configuration revision number would increment by 1 to a new value of 9. SW1 would then flood a VTP subset advertisement across the VTP domain. SW3 and SW4 would add the new VLAN to their VLAN databases. SW2 would ignore this VTP update.

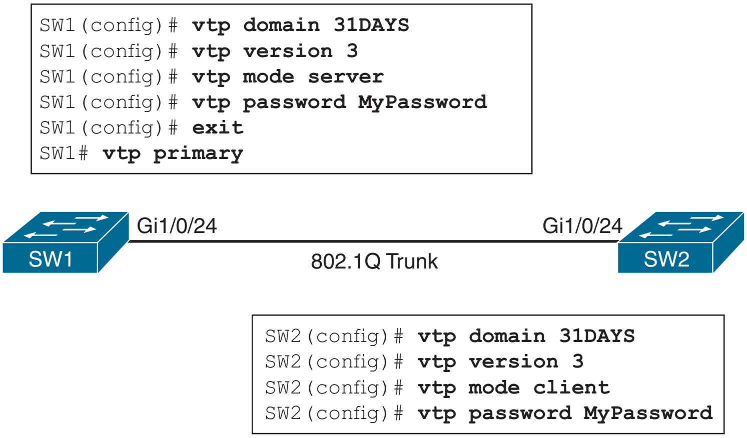
VTP Versions

Three versions of VTP are available for use in a VLAN management domain. Catalyst switches can run either VTP Version 1, 2, or 3. Within a management domain, the versions are not fully interoperable. Therefore, the same VTP version should be configured on every switch in a domain. Switches use VTP Version 1 by default. Most switches now support Version 3, which offers better security, better VLAN database propagation control, MST support, and extended VLAN ranges to 4094. When using Version 3, the primary VTP server must be configured with the **vtp primary** privileged EXEC command.

VTP Configuration Example

Figure 29-9 shows a topology in which SW1 is configured as the VTP Version 3 primary server, and SW2 is configured as the VTP client. Both switches are configured for the same VTP domain (31DAYS) and with the same password.

Figure 29-9 VTP Configuration Example



```
SW1(config)# vtp domain 31DAYS
SW1(config)# vtp version 3
SW1(config)# vtp mode server
SW1(config)# vtp password MyPassword
SW1(config)# exit
SW1# vtp primary
```

```
SW2(config)# vtp domain 31DAYS
SW2(config)# vtp version 3
SW2(config)# vtp mode client
SW2(config)# vtp password MyPassword
```

To verify VTP, use the **show vtp status** command, as shown in Example 29-10.

Example 29-10 Verifying VTP

```
SW1# show vtp status
VTP Version capable              : 1 to 3
VTP version running              : 3
VTP Domain Name                  : 31DAYS
VTP Pruning Mode                 : Disabled
VTP Traps Generation             : Disabled
Device ID                        : acf5.e649.6080

Feature VLAN:
--------------
VTP Operating Mode               : Primary Server
Number of existing VLANs         : 4
Number of existing extended VLANs : 0
Maximum VLANs supported locally  : 4096
Configuration Revision           : 8
```

```
Primary ID                        : acf5.e649.6080
Primary Description               : SW1
MD5 digest                        : 0x12 0x7B 0x0A 0x2C 0x00 0xA6 0xFC 0x05
                                    0x56 0xAA 0x50 0x4B 0xDB 0x0F 0xF7 0x37
<. . . output omitted . . .>
```

```
SW2# show vtp status
VTP Version capable               : 1 to 3
VTP version running               : 3
VTP Domain Name                   : 31DAYS
VTP Pruning Mode                  : Disabled
VTP Traps Generation              : Disabled
Device ID                         : 0062.e24c.c044

Feature VLAN:
--------------
VTP Operating Mode                : Client
Number of existing VLANs          : 4
Number of existing extended VLANs : 0
Maximum VLANs supported locally   : 4096
Configuration Revision            : 8
Primary ID                        : 0062.e24c.c044
Primary Description               : SW2
MD5 digest                        : 0x12 0x7B 0x0A 0x2C 0x00 0xA6 0xFC 0x05
                                    0x56 0xAA 0x50 0x4B 0xDB 0x0F 0xF7 0x37

<. . . output omitted . . .>
```

In the output in Example 29-10, notice that SW1 and SW2 are on the same configuration revision number and have the same number of existing VLANs.

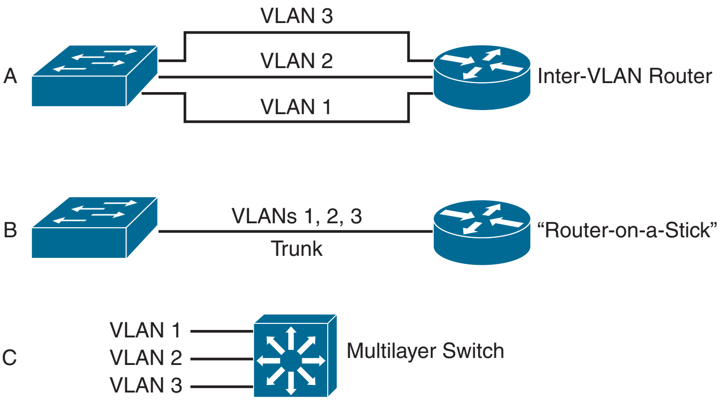
Inter-VLAN Routing

Recall that a Layer 2 network is defined as a broadcast domain. A Layer 2 network can also exist as a VLAN inside one or more switches. VLANs are essentially isolated from each other so that packets in one VLAN cannot cross into another VLAN.

To transport packets between VLANs, you must use a Layer 3 device. Traditionally, this has been a router's function. The router must have a physical or logical connection to each VLAN so that it can forward packets between them. This is known as *inter-VLAN routing*.

Inter-VLAN routing can be performed by an external router that connects to each of the VLANs on a switch. Separate physical connections can be used to achieve this. Part A of Figure 29-10 illustrates this concept. The external router can also connect to the switch through a single trunk link, carrying all the necessary VLANs, as illustrated in Part B of Figure 29-10. Part B illustrates what is commonly referred to as a “router-on-a-stick” because the router needs only a single interface to do its job.

Figure 29-10 Inter-VLAN Routing Models



Finally, Part C of Figure 29-10 shows how the routing and switching functions can be combined into one device: a Layer 3 or multilayer switch. No external router is needed.

Inter-VLAN Routing Using an External Router

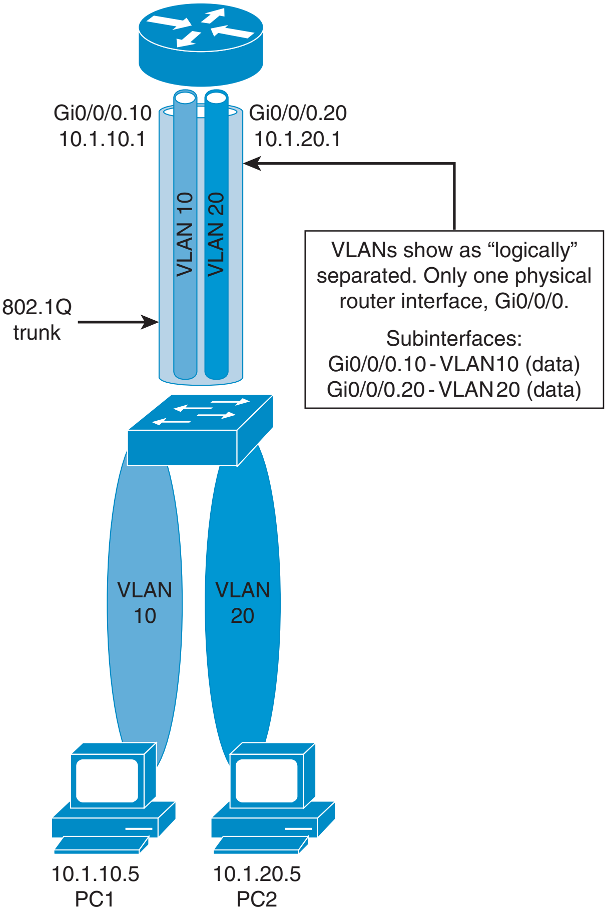
Figure 29-11 shows a configuration in which the router is connected to a switch with a single 802.1Q trunk link. The router can receive packets on one VLAN and forward them to another VLAN. In the example, PC1 can send packets to PC2, which is in a different VLAN. To support 802.1Q trunking, you must subdivide the physical router interface into multiple logical, addressable interfaces—one per VLAN. The resulting logical interfaces are called *subinterfaces*. You associate the VLAN with each subinterface by using the **encapsulation dot1q** *vlan-id* command.

Example 29-11 shows the commands required to configure the router-on-stick illustrated in Figure 29-11.

Example 29-11 Configuring Routed Subinterfaces

```
Router# configure terminal
Router(config)# interface GigabitEthernet 0/0/0.10
Router(config-subif)# encapsulation dot1q 10
Router(config-subif)# ip address 10.0.10.1 255.255.255.0
Router(config-subif)# interface GigabitEthernet 0/0/0.20
Router(config-subif)# encapsulation dot1q 20
Router(config-subif)# ip address 10.0.20.1 255.255.255.0
Router(config-subif)# interface GigabitEthernet 0/0/0.1
Router(config-subif)# encapsulation dot1q 1 native
Router(config-subif)# ip address 10.0.1.1 255.255.255.0
```

Figure 29-11 Inter-VLAN Routing Using an External Router

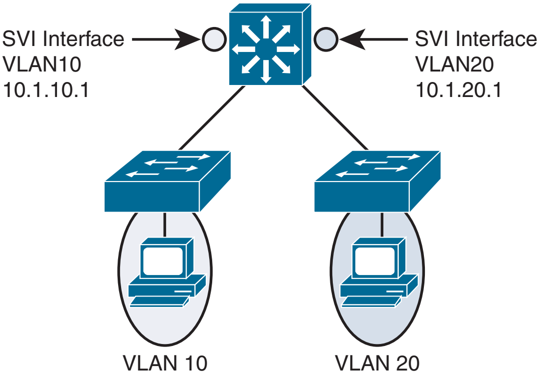

Gi0/0/0.10
10.1.10.1
Gi0/0/0.20
10.1.20.1
VLAN 10
VLAN 20
802.1Q
trunk
VLANs show as “logically” separated. Only one physical router interface, Gi0/0/0.
Subinterfaces:
Gi0/0/0.10 - VLAN10 (data)
Gi0/0/0.20 - VLAN 20 (data)
VLAN
10
VLAN
20
10.1.10.5
PC1
10.1.20.5
PC2

Notice the use of the **native** keyword in Example 29-11. The other option for configuring routing of untagged traffic is to configure the physical interface with the native VLAN IP address. The disadvantage of such a configuration is that when you do not want the untagged traffic to be routed, you must shut down the physical interface, but in doing so, you also shut down all the subinterfaces on that interface.

Inter-VLAN Routing Using Switched Virtual Interfaces

A switched virtual interface (SVI) is a virtual interface that is configured within a multilayer switch. You can create an SVI for any VLAN that exists on the switch. Only one SVI can be associated with one VLAN. An SVI can be configured to operate at Layer 2 or Layer 3, as shown in Figure 29-12. An SVI is virtual in that there is no physical port dedicated to the interface, yet it can perform the same functions for the VLAN as a router interface would. An SVI can be configured in the same way as a router interface (with IP address, inbound or outbound access control lists, and so on). The SVI for the VLAN provides Layer 3 processing for packets to and from all switch ports that are associated with that VLAN.

Figure 29-12 SVI on a Layer 3 Switch



By default, an SVI is created for the default VLAN (VLAN 1) to permit remote switch administration. Additional SVIs must be explicitly created. You create SVIs the first time that you enter the VLAN interface configuration mode for a particular VLAN SVI (for example, when you enter the global configuration command **interface vlan** *vlan-id*). The VLAN number that you use should corresponds to the VLAN tag associated with the data frames on an 802.1Q encapsulated trunk or with the VID that is configured for an access port. You can configure and assign an IP address for each VLAN SVI that is to route traffic from and into a VLAN on a Layer 3 switch.

Example 29-12 shows the commands required to configure the SVIs in Figure 29-12. The example assumes that VLAN 10 and VLAN 20 are already preconfigured.

Example 29-12 Configuring SVIs

```
Switch# configure terminal
Switch(config)# interface vlan 10
Switch(config-if)# ip address 10.0.10.1 255.255.255.0
Switch(config-if)# no shutdown
Switch(config-if)# interface vlan 20
Switch(config-if)# ip address 10.0.20.1 255.255.255.0
Switch(config-if)# no shutdown
```

Routed Switch Ports

A routed switch port is a physical switch port on a multilayer switch that is configured to perform Layer 3 packet processing. You configure a routed switch port by removing the Layer 2 switching capability of the switch port. Unlike an access port or an SVI, a routed port is not associated with a particular VLAN. Also, because Layer 2 functionality has been removed, Layer 2 protocols such as STP and VTP do not function on a routed interface. However, protocols like LACP, which can be used to build either Layer 2 or Layer 3 EtherChannel bundles, still function at Layer 3.

Routed ports are used for point-to-point links; for example, routed ports may be used to connect WAN routers and to connect security devices. In a campus switched network, routed ports are mostly configured between switches in the campus backbone and building distribution switches if

Layer 3 routing is applied at the distribution layer. If Layer 3 routing is deployed at the access layer, then links from access to distribution also use routed switch ports.

To configure routed ports, you configure the respective interface as a Layer 3 interface by using the **no switchport** interface command if the default configurations of the interfaces are Layer 2 interfaces. In addition, you can assign an IP address and other Layer 3 parameters as necessary.

Example 29-13 shows the commands required to configure GigabitEthernet 1/0/23 as a Layer 3 routed switch port.

Example 29-13 Configuring Routed Switch Ports

```
Switch# configure terminal
Switch(config)# interface GigabitEthernet 1/0/23
Switch(config-if)# no switchport
Switch(config-if)# ip address 10.254.254.1 255.255.255.0
```

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	1, 5
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	1, 3

Day 28

Spanning Tree Protocol

ENCOR 350-401 Exam Topics

Infrastructure

- Layer 2
  - Configure and ++verify common Spanning Tree Protocols (RSTP and MST)

Key Topics

Today we review the Layer 2 loop-avoidance mechanism Spanning Tree Protocol (STP), including the configuration, verification, and troubleshooting of Cisco Per-VLAN Spanning Tree (PVST/PVST+), Rapid Spanning Tree Protocol (RSTP), and Multiple Spanning Tree Protocol (MST).

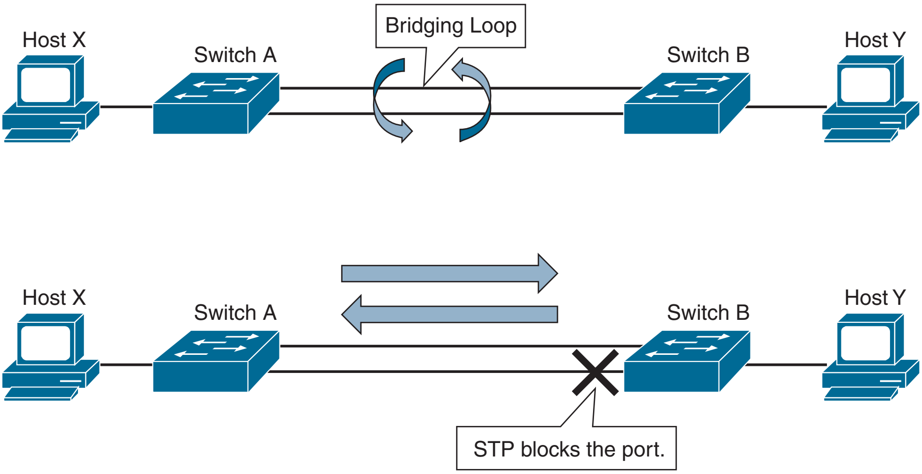
High availability is a primary goal for enterprise networks that rely heavily on their multilayer switched network to conduct business. One way to ensure high availability is to provide Layer 2 redundancy of devices, modules, and links throughout the network. Network redundancy at Layer 2, however, introduces the potential for bridging loops, where frames loop endlessly between devices, crippling the network. STP identifies and prevents such Layer 2 loops. Bridging loops form because parallel switches (or bridges) are unaware of each other. STP was developed to overcome the possibility of bridging loops so that redundant switches and switch paths can be used if a failure occurs. Basically, the protocol enables switches to become aware of each other so they can negotiate a loop-free path through the network.

Older Cisco Catalyst switches use PVST+ by default, and newer switches have Rapid PVST+ enabled instead. Rapid PVST+ is the IEEE 802.1w standard RSTP implemented on a per-VLAN basis. Note that, since 2014, the original IEEE 802.1D standard is now part of the IEEE 802.1Q standard.

IEEE 802.1D STP Overview

Spanning Tree Protocol provides loop resolution by managing the physical paths to given network segments. STP enables physical path redundancy while preventing the undesirable effects of active loops in the network. STP forces certain ports into a blocking state. These blocking ports do not forward data frames, as illustrated in Figure 28-1.

Figure 28-1 Bridging Loop and STP



In a redundant topology, you might see problems such as these:

- **Broadcast storms:** In a broadcast storm, each switch on a redundant network floods broadcast frames endlessly. Switches flood broadcast frames to all ports except the port on which the frame was received. These frames then travel around the loop in all directions.
- **Multiple frame transmission:** Multiple copies of the same unicast frames may be delivered to a destination station, which can cause problems with the receiving protocol.
- **MAC database instability:** This problem results from copies of the same frame being received on different ports of the switch. The MAC address table maps the source MAC address on a received packet to the interface it was received on. If a loop occurs, then the same source MAC address could be seen on multiple interfaces, causing instability.

STP forces certain ports into a standby state so that they do not listen to, forward, or flood data frames. There is only one active path to each network segment. It is a loop-avoidance mechanism that is used to solve problems caused by redundant topology. STP port states are covered later in the chapter.

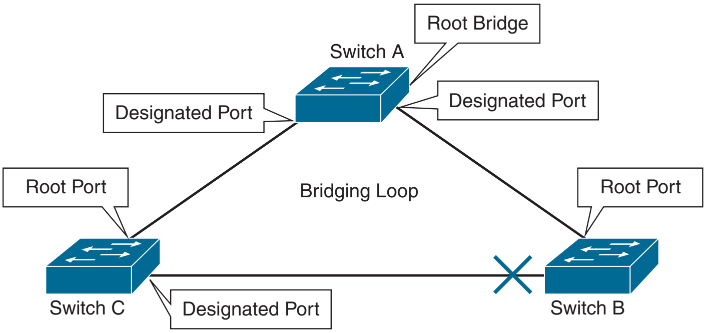
For example, in Figure 28-1, there is a redundant link between Switch A and Switch B. However, this causes a bridging loop. For example, a broadcast or multicast packet that transmits from Host X and is destined for Host Y will continue to loop between these switches. When STP runs on both switches, it blocks one of the ports to prevent the formation of a loop in the network. STP addresses and solves these issues.

To provide this desired path redundancy and to avoid a loop condition, STP defines a tree that spans all the switches in an extended network. STP forces certain redundant data paths into a standby (blocked) state and leaves other paths in a forwarding state. If a link in the forwarding state becomes unavailable, STP reconfigures the network and reroutes data paths through the activation of the appropriate standby path.

STP Operations

STP provides loop resolution by managing the physical path to the given network segment. It does so by performing three steps, as shown in Figure 28-2:

Figure 28-2 STP Operations



1. **Elects one root bridge:** Only one bridge can act as the root bridge. The root bridge is the reference point, and all data flows in the network are from the perspective of this switch. All ports on a root bridge forward traffic.
2. **Selects the root port on each non-root bridge:** One port on each non-root bridge is the root port. It is the port with the lowest-cost path from the non-root bridge to the root bridge. By default, the STP path cost is calculated from the bandwidth of the link. You can also set the STP path cost manually.
3. **Selects the designated port on each segment:** There is one designated port on each segment. It is selected on the bridge with the lowest-cost path to the root bridge and is responsible for forwarding traffic on that segment.

Ports that are neither root nor designated must be non-designated. Non-designated ports are normally in the blocking state to break the loop topology. The overall effect is that only one path to each network segment is active at any time. If there is a problem with connectivity to any of the segments in the network, STP reestablishes connectivity by automatically activating a previously inactive path, if one exists.

Bridge Protocol Data Unit

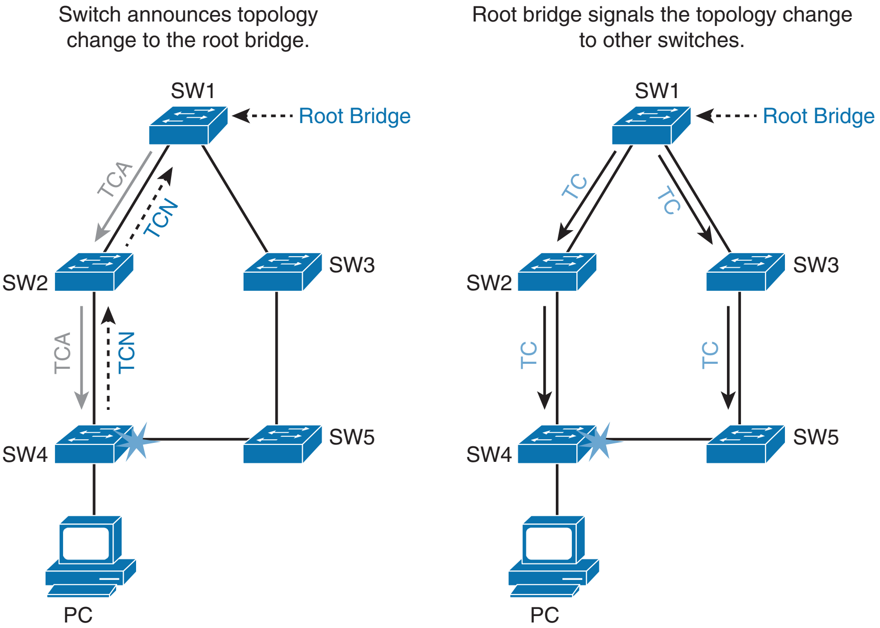
STP uses bridge protocol data units (BPDUs) to exchange STP information specifically for root bridge election and for loop identification. By default, BPDUs are sent out every 2 seconds. BPDUs are generally categorized into three types:

- **Configuration BPDUs:** Used to identify the root bridge, root ports, designated ports, and blocking ports.
- **TCN (topology change notification) BPDUs:** Used when a bridge discovers a change in topology, usually because of a link failure, bridge failure, or port transitioning to the forwarding state. It is forwarded on the root port toward the root bridge.
- **TCA (topology change acknowledgment) BPDUs:** Used by the upstream bridge to respond to the receipt of a TCN.

Every switch sends out BPDUs on each port. The source address is the MAC address of that port, and the destination address is the STP multicast address 01-80-c2-00-00-00.

In normal STP operation, a switch keeps receiving configuration BPDUs from the root bridge on its root port, but it never sends out a BPDU toward the root bridge. When there is a change in topology, such as a new switch being added or a link going down, the switch sends a TCN BPDU on its root port, as shown in Figure 28-3.

Figure 28-3 BPDU TCN Flow


Switch announces topology change to the root bridge.
Root bridge signals the topology change to other switches.
SW1
Root Bridge
TCA
TCN
SW2
SW3
TCA
TCN
SW4
SW5
PC
SW1
Root Bridge
TC
TC
SW2
SW3
TC
TC
SW4
SW5
PC

The designated switch receives the TCN, acknowledges it, and generates another one for its own root port. The process continues until the TCN hits the root bridge. The designated switch acknowledges the TCN by immediately sending back a normal configuration BPDU with the TCA bit set. The switch that notifies the topology change does not stop sending its TCN until the designated switch has acknowledged it. Therefore, the designated switch answers the TCN even though it has not yet received a configuration BPDU from its root.

Once the root is aware that there has been a topology change event in the network, it starts to send out its configuration BPDUs with the topology change (TC) bit set. These BPDUs are relayed by every bridge in the network with this bit set. Bridges receive topology change BPDUs on both forwarding and blocking ports.

There are three types of topology change:

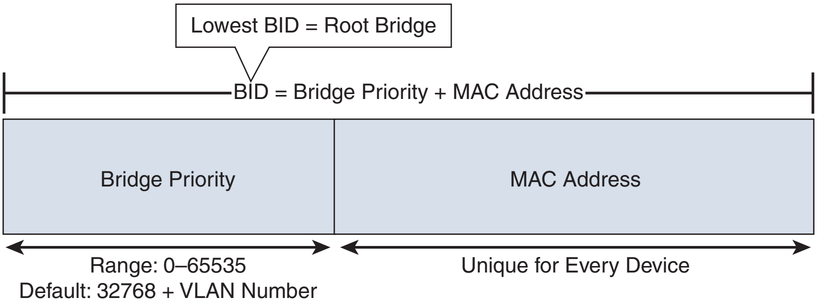
- **Direct topology change:** This type of change can be detected on an interface. In Figure 28-3, SW4 has detected a link failure on one of its interfaces. It then sends out a TCN message on the root port to reach the root bridge. SW1, the root bridge, announces the topology change to other switches in the network. All switches shorten their bridging table aging time to the forward delay (15 seconds). This way, they get new port and MAC address associations after 15 seconds, not after 300 seconds, which is the default bridging table aging time. The convergence time in that case is two times the forward delay period—that is, 30 seconds.

- **Indirect topology change:** With this type of change, the link status stays up. Something (for example, another device such as firewall) on the link has failed or is filtering traffic, and no data is received on each side of the link. Because there is no link failure, no TCN messages are sent. The topology change is detected because there are no BPDUs from the root bridge. With an indirect link failure, the topology does not change immediately, but STP converges again, thanks to timer mechanisms. The convergence time in that case is longer than with direct topology change. First, because of the loss of BPDU, the Max Age timer has to expire (20 seconds). Then the port transitions to listening (15 seconds) and then learning (15 seconds), for a total of 50 seconds.
- **Insignificant topology change:** This type of change occurs if, for example, a PC connected to SW4 is turned off. This event causes SW4 to send out TCNs. However, because none of the switches had to change port states to reach the root bridge, no actual topology change occurred. The only consequence of shutting down the PC is that all switches age out entries from the content-addressable memory (CAM) table sooner than normal. This can become a problem if you have a large number of PCs. Many PCs going up and down can cause a substantial number of TCN exchanges. To avoid this, you can enable PortFast on end-user ports. If a PortFast-enabled port goes up or down, a TCN is not generated.

Root Bridge Election

For all switches in a network to agree on a loop-free topology, a common frame of reference must exist to use as a guide. This reference point is called the root bridge. (The term *bridge* continues to be used even in a switched environment because STP was developed for use in bridges.) An election process among all connected switches chooses the root bridge. Each switch has a unique bridge ID (BID) that identifies it to other switches. The BID is an 8-byte value consisting of two fields, as shown in Figure 28-4:

Figure 28-4 STP Bridge ID



- **Bridge Priority** (2 bytes)**:** The priority or weight of a switch in relationship to all other switches. The Bridge Priority field can have a value of 0 to 65,535, and it defaults to 32,768 (or 0x8000) on every Catalyst switch. In PVST and PVST+ implementations of STP, the original 16-bit Bridge Priority field is split into two fields, resulting in the following components in the BID:

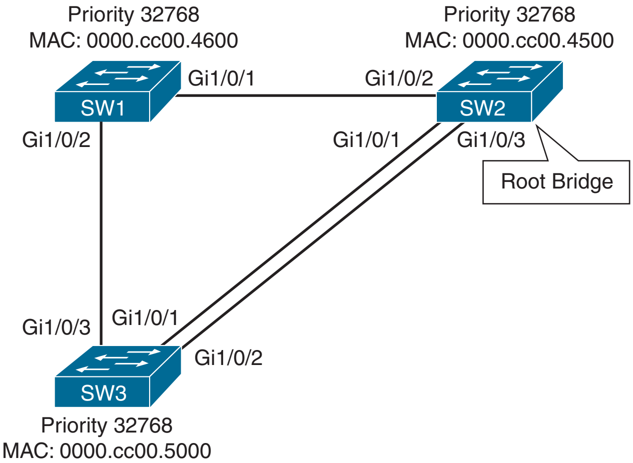
  - **Bridge Priority:** A 4-bit field used to carry bridge priority. The default priority is 32,768, which is the midrange value. The priority is conveyed in discrete values in increments of 4096.
  - **Extended System ID:** A 12-bit field carrying the VLAN ID. This ensures a unique BID for each VLAN configured on the switch.
- **MAC Address** (6 bytes)**:** The MAC address used by a switch can come from the Supervisor module, the backplane, or a pool of 1024 addresses that is assigned to every supervisor or backplane, depending on the switch model. In any event, this address is hard-coded and unique, and the user cannot change it.

The root bridge is selected based on the lowest BID. If all switches in the network have the same priority, the switch with the lowest MAC address becomes the root bridge.

In the beginning, each switch assumes that it is the root bridge. Each switch sends a BPDU to its neighbors, presenting its BID. At the same time, it receives BPDUs from all its neighbors. Each time a switch receives a BPDU, it checks that BID against its own. If the received bridge ID is better than its own, the switch realizes that it, itself, is not the root bridge. Otherwise, it keeps the assumption of being the root bridge.

Eventually, the process converges, and all switches agree that one of them is the root bridge, as illustrated in Figure 28-5.

Figure 28-5 STP Root Bridge Election



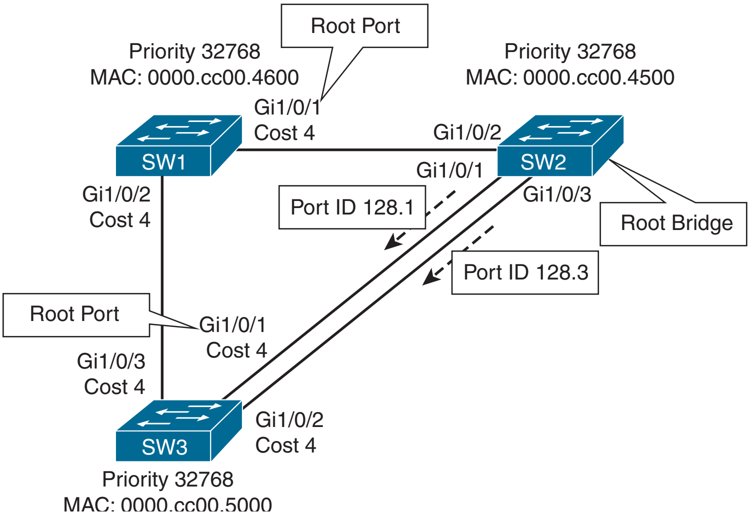
Root bridge election is an ongoing process. If a new switch appears with a better BID, it is elected the new root bridge. STP includes mechanisms to protect against random or undesirable root bridge changes.

Root Port Election

After the root bridge is elected, each non-root bridge must figure out where it is in relationship to the root bridge. The root port is the port with the best path to the root bridge. To determine root ports on non-root bridges, cost value is used. The path cost is the cumulative cost of all links to the root bridge. The root port has the lowest cost to the root bridge. If two ports have the same cost, the sender's port ID is used to break the tie.

In Figure 28-6, SW1 has two paths to the root bridge. The root path cost is a cumulative value. The cost of link SW1–SW2 is 4, and the cost of link SW3–SW2 is also 4. The cumulative cost of the path SW1–SW3–SW2 through Gi1/0/2 is 4 + 4 = 8, whereas the cumulative cost of the link SW1–SW2 through Gi1/0/1 is 4. Since the path through GigabitEthernet 1/0/1 has a lower cost, GigabitEthernet 1/0/1 is elected the root port.

Figure 28-6 STP Root Port Election


Root Port
Priority 32768
MAC: 0000.cc00.4600
Priority 32768
MAC: 0000.cc00.4500
Gi1/0/1
Cost 4
Gi1/0/2
SW1
SW2
Gi1/0/1
Gi1/0/2
Cost 4
Port ID 128.1
Gi1/0/3
Root Bridge
Port ID 128.3
Root Port
Gi1/0/1
Cost 4
Gi1/0/3
Cost 4
SW3
Gi1/0/2
Cost 4
Priority 32768
MAC: 0000.cc00.5000

When two ports have the same cost, arbitration can be done using the advertised port ID (from the neighboring switch). In Figure 28-6, SW3 has three paths to the root bridge. Through Gi1/0/3, the cumulative cost is 8 (links SW3–SW1 and SW1–SW2). Through Gi1/0/1 and Gi1/0/2, the cost is 4. Because lower cost is better, one of these two ports will be elected the root port. Port ID is a combination of a port priority, which is 128 by default, and a port number. For example, in Figure 28-6, the port Gi1/0/1 on SW2 has the port ID 128.1; the port Gi1/0/3 has port ID 128.3. The lowest port ID is always chosen when port ID is the determining factor. Because Gi1/0/1 receives a lower port ID from SW2 (128.1) than Gi1/0/2 receives (128.3), Gi1/0/1 is elected the root port.

STP cost is calculated from the bandwidth of the link. It can be manually changed by the administrator, although such changes are not commonly made.

Table 28-1 shows common link cost values. The higher the bandwidth of a link, the lower the cost of transporting data across it. Cisco Catalyst switches support two STP path cost modes: short mode and long mode. Short mode is based on a 16-bit value with a link speed reference value of 20 Gbps, whereas long mode uses a 32-bit value with a link speed reference value of 20 Tbps. To set either short or long mode path cost calculation, use the **spanning-tree pathcost method** command.

Table 28-1 Default Interface STP Port Costs

Link Speed	Short-Mode STP Cost	Long-Mode STP Cost
10 Mbps	100	2,000,000
10 Mbps	19	200,000
1 Gbps	4	20,000
10 Gbps	2	2000

Designated Port Election

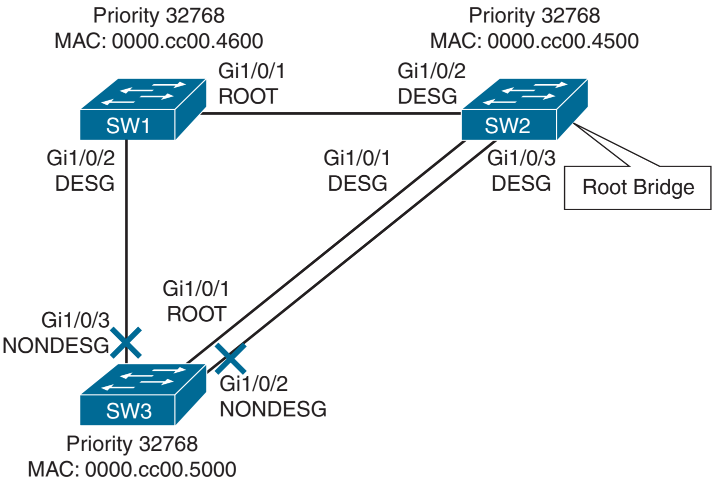
After the root bridge and root ports on non-root bridges have been elected, STP has to identify which port on the segment should forward the traffic in order to prevent loops from occurring in the network. Only one of the ports on a segment should forward traffic to and from that segment. The designated port—the one forwarding the traffic—is also chosen based on the lowest cost to the root bridge. On the root bridge, all ports are designated.

If there are two paths to the root bridge with equal cost, STP uses the following criteria for best path determination and consequently for determining the designated and non-designated ports on the segment:

- Lowest root path cost to the root bridge
- Lowest sender BID
- Lowest sender port ID

As shown in Figure 28-7, SW2 is the root bridge, so all its ports are designated. To prevent loops, a blocking port for the SW1–SW3 segment has to be determined. Because SW3 and SW1 have the same path cost to the root bridge, 4, the lower BID breaks the tie. SW1 has a lower BID than SW3, so the designated port for the segment is GigabitEthernet1/0/2 on SW1.

Figure 28-7 STP Designated Port Election


Priority 32768
MAC: 0000.cc00.4600
Priority 32768
MAC: 0000.cc00.4500
Gi1/0/1
ROOT
Gi1/0/2
DESG
SW1
SW2
Gi1/0/2
DESG
Gi1/0/1
DESG
Gi1/0/3
DESG
Root Bridge
Gi1/0/1
ROOT
Gi1/0/3
NONDESG
Gi1/0/2
NONDESG
SW3
Priority 32768
MAC: 0000.cc00.5000

Only one port on a segment should forward traffic. All ports that are not root or designated ports are non-designated ports. Non-designated ports go to the blocking state to prevent a loop. Non-designated ports are also referred to as *alternate*, or *backup*, *ports*.

In Figure 28-7, root ports and designated ports are determined on non-root bridges. All the other ports are non-designated. The only two interfaces that are not root or designated ports are GigabitEthernet1/0/2 and GigabitEthernet1/0/3 on SW3. Both are non-designated (blocking).

STP Port States

To participate in the STP process, a switch port must go through several states. A port starts in the disabled state, and then, after an administrator enables it, it moves through various states until it reaches the forwarding state if it is a designated port or a root port. If not, it is moved into the blocking state. Table 28-2 outlines all the STP states and their functionality:

Table 28-2 STP Port States

STP Port State	Receives BPDU	Sends BPDU	Learns MAC Addresses	Forwards Data Packets	Duration
Disabled	No	No	No	No	Until administrator enables port
Blocking	Yes	No	No	No	Undefined
Listening	Yes	Yes	No	No	Forward delay (15 seconds)
Learning	Yes	Yes	Yes	No	Forward delay (15 seconds)
Forwarding	Yes	Yes	Yes	Yes	Undefined

- **Blocking:** In this state, a port ensures that no bridging loops occur. A port in this state cannot receive or transmit data, but it receives BPDUs, so the switch can hear from its neighbor switches and determine the location and root ID of the root switch and the port role of each switch. A port in this state is a non-designated port, so it does not participate in active topology.
- **Listening:** A port is moved from the blocking state to the listening state if there is a possibility that it will be selected as the root or designated port. A port in this state cannot send or receive data frames, but it is allowed to send and receive BPDUs, so it participates in the active Layer 2 topology.
- **Learning:** After the listening state expires (in 15 seconds), the port is moved to the learning state. The port sends and receives BPDUs, and it can also learn and add new MAC addresses to its table. A port in this state cannot send any data frames.

- **Forwarding:** After the learning state expires (in 15 seconds), the port is moved to the forwarding state if it is to become a root or designated port. It is now considered part of the active Layer 2 topology. It sends and receives frames and sends and receives BPDUs.
- **Disabled:** In this state, a port is administratively shut down. It does not participate in STP and does not forward frames.

Rapid Spanning Tree Protocol

Rapid Spanning Tree Protocol (RSTP; specified in IEEE 802.1w) significantly speeds the recalculation of the spanning tree when the network topology changes. RSTP defines the additional port roles alternate and backup and defines port states as discarding, learning, or forwarding.

RSTP is an evolution, rather than a revolution, of the 802.1D standard. The 802.1D terminology remains primarily the same, and most parameters are left unchanged. On Cisco Catalyst switches, a rapid version of PVST+, called RPVST+ or PVRST+, is the per-VLAN version of the RSTP implementation. All the current-generation Catalyst switches support Rapid PVST+, and it is now the default version enabled on Catalyst 9000 Series switches.

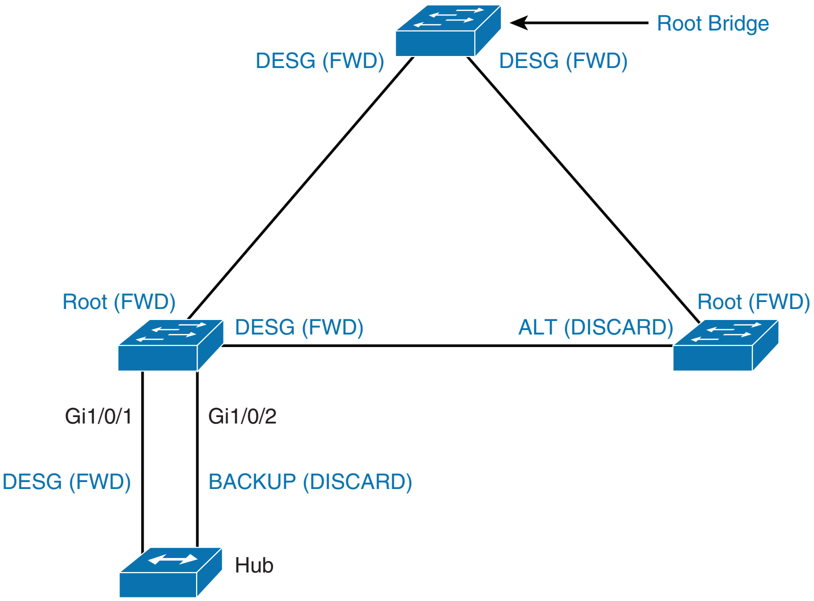
RSTP Port Roles

The port role defines the ultimate purpose of a switch port and the way it handles data frames. RSTP port roles differ slightly from STP roles. RSTP defines the following port roles (and Figure 28-8 illustrates the port roles in a three-switch topology):

- **Root:** The root port is the switch port on every non-root bridge that is the chosen path to the root bridge. There can be only one root port on every non-root switch. The root port is considered part of the active Layer 2 topology. It forwards, sends, and receives BPDUs (data messages).
- **Designated:** Each switch has at least one switch port as the designated port for a segment. In the active Layer 2 topology, the switch with the designated port receives frames on the segment that are destined for the root bridge. There can be only one designated port per segment.
- **Alternate:** The alternate port is a switch port that offers an alternate path toward the root bridge. It assumes a discarding state in an active topology. The alternate port makes a transition to a designated port if the current designated path fails.
- **Disabled:** A disabled port has no role in the operation of spanning tree.
- **Backup:** The backup port is an additional switch port on the designated switch with a redundant link to a shared segment for which the switch is designated. The backup port has the discarding state in the active topology.

Notice that instead of the STP non-designated port role, there are now alternate and backup ports. These additional port roles allow RSTP to define a standby switch port before a failure or topology change. The alternate port moves to the forwarding state if there is a failure on the designated port for the segment. A backup port is used only when a switch is connected to a shared segment using a hub, as illustrated in Figure 28-8.

Figure 28-8 RSTP Port Roles



RSTP Port States

The RSTP port states correspond to the three basic operations of a switch port: discarding, learning, and forwarding. There is no listening state with RSTP, as there is with STP. With RSTP, the listening and blocking STP states are replaced with the discarding state. In a stable topology, RSTP ensures that every root port and designated port transit to the forwarding state, and all alternate ports and backup ports are always in the discarding state. Table 28-3 depicts the characteristics of RSTP port states.

Table 28-3 RSTP Port States

RSTP Port State	Description
Discarding	This state is seen in both a stable active topology and during topology synchronization and changes. The discarding state prevents the forwarding of data frames, thus "breaking" the continuity of a Layer 2 loop.
Learning	This state is seen in both a stable active topology and during topology synchronization and changes. The learning state accepts data frames to populate the MAC table to limit flooding of unknown unicast frames. Data frames are not forwarded.
Forwarding	This state is seen only in stable active topologies. The forwarding switch ports determine the topology. Following a topology change or during synchronization, the forwarding of data frames occurs only after a proposal-and-agreement process.

A port accepts and processes BPDU frames in all port states.

RSTP Rapid Transition to Forwarding State

A quick transition to the forwarding state is a key feature of 802.1w. The legacy STP algorithm passively waited for the network to converge before it turned a port into the forwarding state. To achieve faster convergence, a network administrator had to manually tune the conservative default parameters (Forward Delay and Max Age timers). This often put the stability of the network at stake. RSTP is able to quickly confirm that a port can safely transition to the forwarding state without having to rely on any manual timer configuration. In order to achieve fast convergence on a port, the protocol relies on two new variables: edge ports and link type.

Edge Ports

The edge port concept is already well known to Cisco STP users, as it basically corresponds to the PortFast feature. Ports that are directly connected to end stations cannot create bridging loops in the network. An edge port directly transitions to the forwarding state and skips the listening and learning stages. Neither edge ports nor PortFast-enabled ports generate topology changes when the link toggles. An edge port that receives a BPDU immediately loses edge port status and becomes a normal STP port. Cisco maintains that the PortFast feature can be used for edge port configuration in RSTP.

Link Type

RSTP can achieve rapid transition to the forwarding state only on edge ports and on point-to-point links. The link type is automatically derived from the duplex mode of a port. A port that operates in full-duplex is assumed to be point-to-point, while a half-duplex port is considered to be a shared port by default. This automatic link type setting can be overridden with explicit configuration. In switched networks today, most links operate in full-duplex mode and are treated as point-to-point links by RSTP. This makes them candidates for rapid transition to the forwarding state.

RSTP Synchronization

To participate in RSTP convergence, a switch must decide the state of each of its ports. Non-edge ports begin in the discarding state. After BPDUs are exchanged between the switch and its neighbor, the root bridge can be identified. If a port receives a superior BPDU from a neighbor, that port becomes the root port.

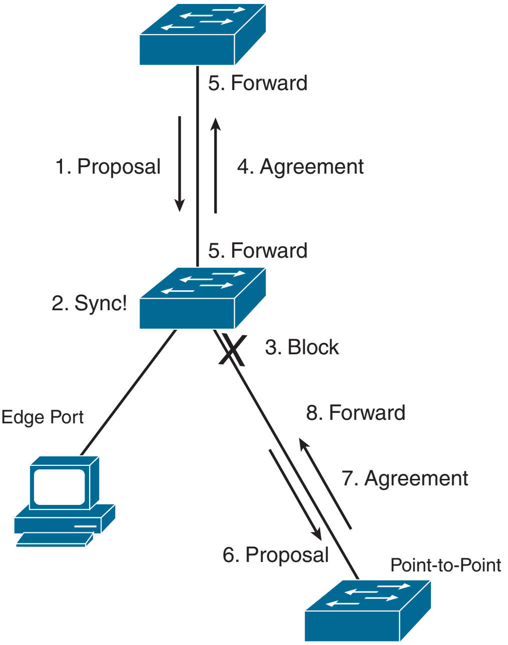
For each non-edge port, the switch exchanges a proposal-agreement handshake to decide the state of each end of the link. Each switch assumes that its port should become the designated port for the segment, and a proposal message (a configuration BPDU) is sent to the neighbor to suggest this.

When a switch receives a proposal message on a port, the following sequence of events occurs (and Figure 28-9 shows the sequence, based on the center switch):

1. If the proposal's sender has a superior BPDU, the local switch realizes that the sender should be the designated switch (having the designated port) and that its own port must become the new root port.
2. Before the switch agrees to anything, it must synchronize itself with the topology.
3. All non-edge ports immediately are moved into the discarding (blocking) state so that no bridging loops can form.

4. An agreement message (a configuration BPDU) is sent back to the sender, indicating that the switch agrees with the new designated port choice. This also tells the sender that the switch is in the process of synchronizing itself.
5. The root port immediately is moved to the forwarding state. The sender's port also immediately can begin forwarding.
6. For each non-edge port that is currently in the discarding state, a proposal message is sent to the respective neighbor.
7. An agreement message is expected and received from a neighbor on a non-edge port.
8. The non-edge port is immediately moved to the forwarding state.

Figure 28-9 RSTP Convergence

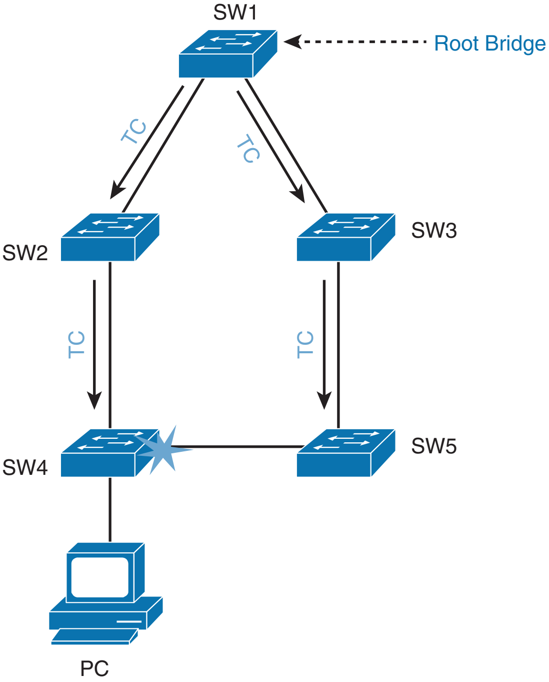


Notice that the RSTP convergence begins with a switch sending a proposal message. The recipient of the proposal must synchronize itself by effectively isolating itself from the rest of the topology. All non-edge ports are blocked until a proposal message can be sent, causing the nearest neighbors to synchronize themselves. This creates a moving "wave" of synchronizing switches, which can quickly decide to start forwarding on their links only if their neighbors agree.

RSTP Topology Change

For RSTP, a topology change occurs only when a non-edge port transitions to the forwarding state. This means a loss of connectivity is not considered a topology change, as it is with STP. A switch announces a topology change by sending out BPDUs with the TC bit set from all the non-edge designated ports. This way, all the neighbors are informed about the topology change, and they can correct their bridging tables. In Figure 28-10, SW4 sends BPDUs out all its non-edge ports after it detects a link failure. SW2 then sends the BPDU to all its neighbors except for the one that received the BPDU from SW4, and so on.

Figure 28-10 RSTP Topology Change

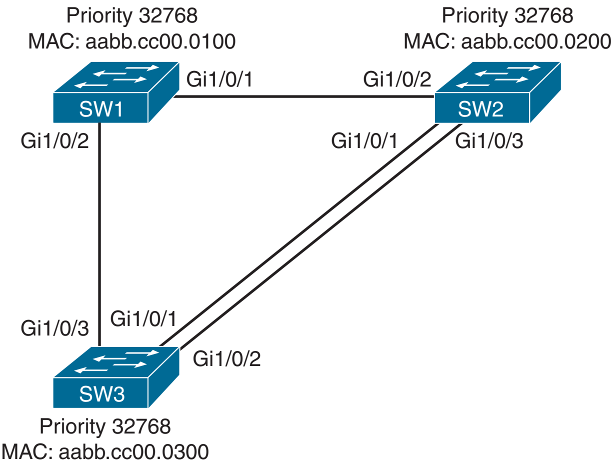


When a switch receives a BPDU with TC bit set from a neighbor, it clears the MAC addresses learned on all its ports except the one that receives the topology change. The switch also receives BPDUs with the TC bit set on all designated ports and the root port. RSTP no longer uses the specific TCN BPDUs unless a legacy bridge needs to be notified. With RSTP, the TC propagation is a one-step process. In fact, the initiator of the topology change floods this information throughout the network, whereas with 802.1D, only the root does. This mechanism is much faster than the 802.1D equivalent.

STP and RSTP Configuration and Verification

Using the topology shown in Figure 28-11, this section reviews how to manually configure a root bridge and the path for spanning tree. In the topology, all switches are initially configured with PVST+ and are in VLAN 1. This configuration example also allows you to verify STP and RSTP functionality.

Figure 28-11 STP/RSTP Configuration Example Topology


Priority 32768
MAC: aabb.cc00.0100
Priority 32768
MAC: aabb.cc00.0200
SW1
Gi1/0/1
Gi1/0/2
SW2
Gi1/0/2
Gi1/0/1
Gi1/0/3
Gi1/0/3
Gi1/0/1
Gi1/0/2
SW3
Priority 32768
MAC: aabb.cc00.0300

There are two loops in this topology: SW1–SW2–SW3 and SW2–SW3. Wiring the network in this way provides redundancy, but Layer 2 loops occur if STP does not block redundant links. By default, STP is enabled on all the Cisco switches for VLAN 1. To find out which switch is the root switch and discover the STP port role for each switch, use the **show spanning-tree** command, as shown in Example 28-1.

Example 28-1 Verifying the STP Bridge ID

```
SW1# show spanning-tree

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority     32769
              Address      aabb.cc00.0100
              This bridge is the root
              Hello Time    2 sec   Max Age 20 sec   Forward Delay 15 sec

  Bridge ID   Priority     32769   (priority 32768 sys-id-ext 1)
              Address      aabb.cc00.0100
<... output omitted ...>
```

```
SW2# show spanning-tree

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority     32769
              Address      aabb.cc00.0100
              Cost          100
```

```
             Port        3 (GigabitEthernet1/0/2)
             Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32769  (priority 32768 sys-id-ext 1)
             Address     aabb.cc00.0200
<... output omitted ...>
```

```
SW3# show spanning-tree

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID    Priority    32769
             Address     aabb.cc00.0100
             Cost        100
             Port        4 (GigabitEthernet1/0/3)
             Hello Time  2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32769  (priority 32768 sys-id-ext 1)
             Address     aabb.cc00.0300
<... output omitted ...>
```

SW1 is the root bridge. Because all three switches have the same bridge priority (32769), the switch with the lowest MAC address is elected as the root bridge. Recall that the default bridge priority is 32768, but the extended system ID value for VLAN 1 is added, giving us 32769.

The first line of output for each switch confirms that the active spanning tree protocol is the IEEE-based PVST+.

Using the **show spanning-tree** command allows you to investigate the port roles on all three switches, as shown in Example 28-2.

Example 28-2 Verifying the STP Port Roles

```
SW1# show spanning-tree
<... output omitted ...>
Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Gi1/0/1             Desg FWD 4         128.1    P2p
Gi1/0/2             Desg FWD 4         128.2    P2p
```

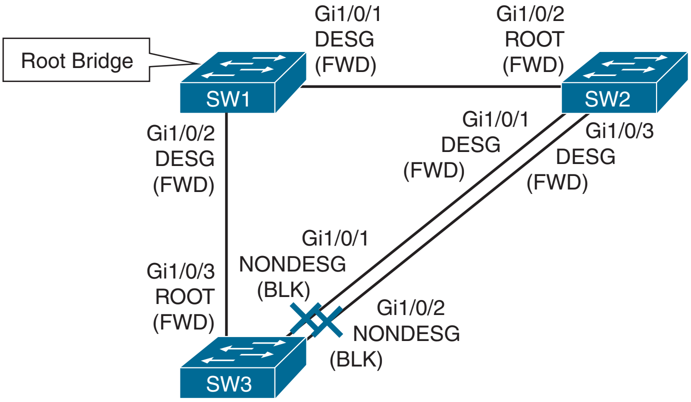
```
SW2# show spanning-tree
<... output omitted ...>
Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Gi1/0/1             Desg FWD 4         128.1    P2p
Gi1/0/2             Root FWD 4         128.2    P2p
Gi1/0/3             Desg FWD 4         128.3    P2p
```

```
SW3# show spanning-tree
<... output omitted ...>
Interface            Role Sts Cost       Prio.Nbr Type
-------------------  ---- --- ---------  -------- --------------------------------
Gi1/0/1              Altn BLK 4          128.1    P2p
Gi1/0/2              Altn BLK 4          128.2    P2p
Gi1/0/3              Root FWD 4          128.3    P2p
```

Because SW1 is the root bridge, it has both of its connected ports in the designated (forwarding) state.

Because SW2 and SW3 are not the root bridge, only one port must be elected root on each of these two switches. The root port is the port with the lowest cost to the root bridge. As SW2 has a lower BID than SW3, all ports on SW2 are set to designated. Other ports on SW3 are non-designated. The Cisco-proprietary protocol PVST+ uses the term "alternate" for non-designated ports. Figure 28-12 shows the summary of the spanning-tree topology and the STP port states for the three-switch topology.

Figure 28-12 STP Port Roles and States



Changing STP Bridge Priority

It is not advisable for a network to choose the root bridge by itself. If all switches have default STP priorities, the switch with the lowest MAC address becomes the root bridge. The oldest switch has the lowest MAC address because the lower MAC addresses were factory assigned first. To manually set the root bridge, you can change a switch's bridge priority. In Figure 28-12, assume that the access layer switch SW3 becomes the root bridge because it has the oldest MAC address. If SW3 is the root bridge, the link between the distribution layer switches is blocked. The traffic between SW1 and SW2 then needs to go through SW3, which is not optimal.

The priority can be a value between 0 and 65,535, in increments of 4096.

A better solution is to use the **spanning-tree vlan** *vlan-id* **root** {**primary** | **secondary**} command. This command is actually a macro that lowers the switch's priority number so that it becomes the root bridge.

To configure the switch to become the root bridge for a specified VLAN, use the **primary** keyword. Use the **secondary** keyword to configure a secondary root bridge. This prevents the slowest and oldest access layer switch from becoming the root bridge if the primary root bridge fails.

The **spanning-tree root** command calculates the priority by learning the current root priority and lowering its priority by 4096. For example, if the current root priority is more than 24,576, the local switch sets its priority to 24,576. If the root bridge has priority lower than 24,576, the local switch sets its priority to 4096 less than the priority of the current root bridge. Configuring the secondary root bridge sets a priority of 28,672. There is no way for the switch to figure out what is the second-best priority in the network. So, setting the secondary priority to 28,672 is just a best guess. It is also possible to manually enter a priority value by using the **spanning-tree vlan** *vlan-id* **priority** *bridge-priority* configuration command.

If you issue the **show running-configuration** command, the output shows the switch's priority as a number (not the **primary** or **secondary** keyword).

Example 28-3 shows the command to make SW2 the root bridge and the output from the **show spanning-tree** command to verify the result.

Example 28-3 Configuring STP Root Bridge Priority

```
SW2(config)# spanning-tree vlan 1 root primary
```

```
SW2# show spanning-tree

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority    24577
              Address     aabb.cc00.0200
              This bridge is the root
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    28673  (priority 28672 sys-id-ext 1)
              Address     aabb.cc00.0200
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  15  sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Gi1/0/1             Desg FWD 4         128.1    P2p
Gi1/0/2             Desg FWD 4         128.2    P2p
Gi1/0/3             Desg FWD 4         128.3    P2p
```

```
SW1# show spanning-tree
<... output omitted ...>
```

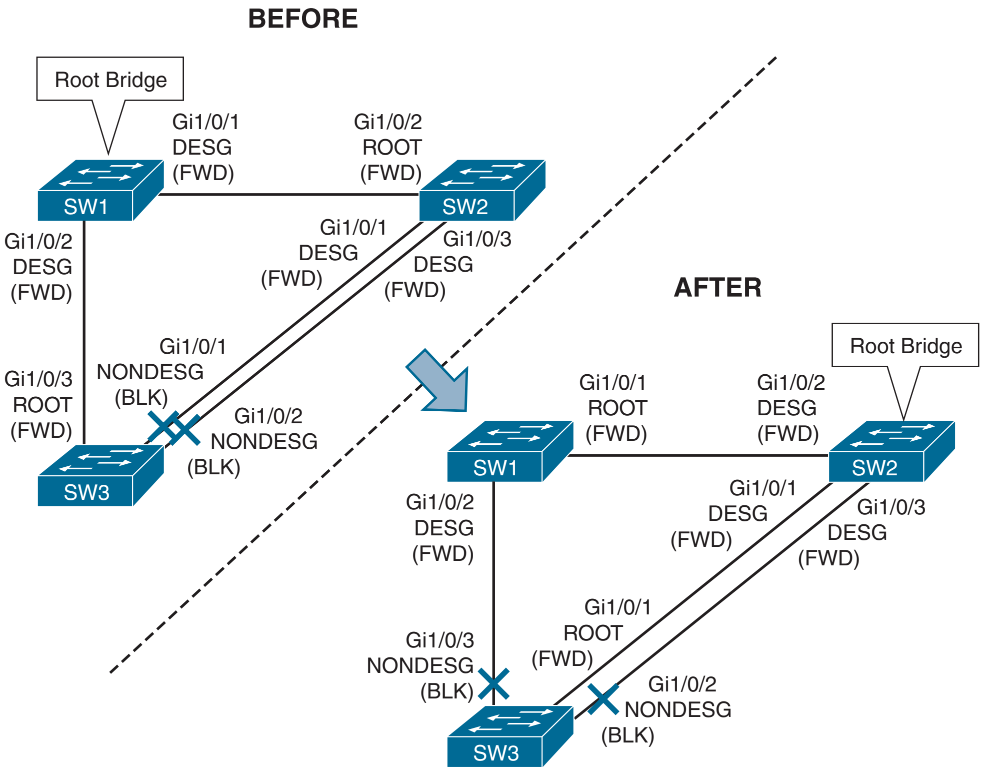
```
Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Gi1/0/1             Root FWD 4         128.1    P2p
Gi1/0/2             Desg FWD 4         128.2    P2p
```

```
SW3# show spanning-tree
<... output omitted ...>
Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Gi1/0/1             Root FWD 4         128.1    P2p
Gi1/0/2             Altn BLK 4         128.2    P2p
Gi1/0/3             Altn BLK 4         128.3    P2p
```

Since SW2 is the root bridge, all its ports are in the designated state, or forwarding. SW1 and SW3 have changed port roles according to the change of the root bridge.

Figure 28-13 shows the port roles before and after you configure SW2 as the root bridge.

Figure 28-13 Root Bridge Change from SW1 to SW2


BEFORE
Root Bridge
Gi1/0/1 DESG (FWD)
Gi1/0/2 ROOT (FWD)
SW1
SW2
Gi1/0/2 DESG (FWD)
Gi1/0/1 DESG (FWD)
Gi1/0/3 DESG (FWD)
Gi1/0/1 NONDESG (BLK)
Gi1/0/3 ROOT (FWD)
Gi1/0/2 NONDESG (BLK)
SW3
AFTER
Root Bridge
Gi1/0/1 ROOT (FWD)
Gi1/0/2 DESG (FWD)
SW1
SW2
Gi1/0/2 DESG (FWD)
Gi1/0/1 DESG (FWD)
Gi1/0/3 DESG (FWD)
Gi1/0/1 ROOT (FWD)
Gi1/0/3 NONDESG (BLK)
Gi1/0/2 NONDESG (BLK)
SW3

STP Path Manipulation

For port role determination, the cost value is used. If all ports have the same cost, the sender's port ID breaks the tie. To control active port selection, change the cost of the interface or the sender's interface port ID.

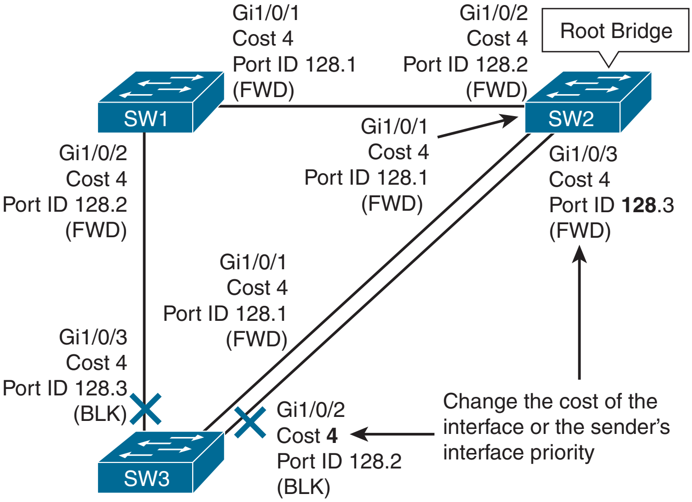
You can modify port cost by using the **spanning-tree vlan** *vlan-id* **cost** *cost-value* command. The cost value can be between 1 and 65,535.

The port ID consists of a port priority and a port number. The port number is fixed because it is based only on its hardware location, but you can influence the port ID by configuring the port priority.

You modify the port priority by using the **spanning-tree vlan** *vlan-id* **port-priority** *port-priority* command. The value of port priority can be between 0 and 255; the default is 128. A lower port priority means a more preferred path to the root bridge.

As shown in Figure 28-14, GigabitEthernet1/0/1 and GigabitEthernet1/0/2 of SW3 have the same interface STP cost to the root SW2. GigabitEthernet1/0/1 of SW3 is forwarding because its sender's port ID of GigabitEthernet1/0/1 of SW2 (128.1) is lower than that of its GigabitEthernet1/0/3 (128.3) of SW2. One way that you could transition SW3's GigabitEthernet1/0/2 to the forwarding port state is to lower the port cost on GigabitEthernet1/0/2. Another way to transition SW3's GigabitEthernet1/0/2 port state to forwarding is to lower the sender's port priority. In this case, this is GigabitEthernet1/0/3 on SW2.

Figure 28-14 STP Path Manipulation

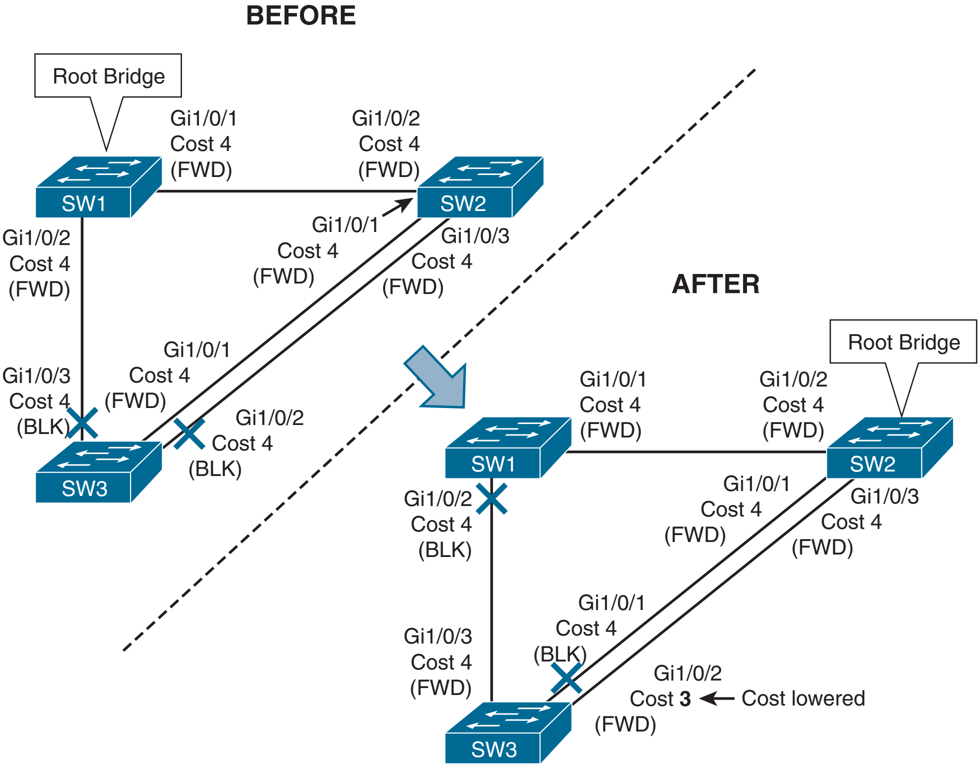

Gi1/0/1
Cost 4
Port ID 128.1
(FWD)
Gi1/0/2
Cost 4
Port ID 128.2
(FWD)
Root Bridge
SW1
SW2
Gi1/0/1
Cost 4
Port ID 128.1
(FWD)
Gi1/0/2
Cost 4
Port ID 128.2
(FWD)
Gi1/0/3
Cost 4
Port ID **128**.3
(FWD)
Gi1/0/1
Cost 4
Port ID 128.1
(FWD)
Gi1/0/3
Cost 4
Port ID 128.3
(BLK)
Gi1/0/2
Cost **4**
Port ID 128.2
(BLK)
SW3
Change the cost of the
interface or the sender's
interface priority

Example 28-4 shows that by changing the cost of SW3's GigabitEthernet1/0/2 interface, the sender interface port priority is no longer observed. STP checks port priority only when costs are equal. Figure 28-15 shows the topology before and after manipulation of the STP port cost.

Example 28-4 Configuration to Change the STP Port Cost

```
SW3(config)# interface GigabitEthernet 1/0/2
SW3(config-if)# spanning-tree vlan 1 cost 3
```

Figure 28-15 STP Interface Cost Manipulation


BEFORE
Root Bridge
SW1
SW2
SW3
Gi1/0/1
Cost 4
(FWD)
Gi1/0/2
Cost 4
(FWD)
Gi1/0/2
Cost 4
(FWD)
Gi1/0/1
Cost 4
(FWD)
Gi1/0/3
Cost 4
(FWD)
Gi1/0/3
Cost 4
(BLK)
Gi1/0/1
Cost 4
(FWD)
Gi1/0/2
Cost 4
(BLK)
AFTER
Root Bridge
SW1
SW2
SW3
Gi1/0/1
Cost 4
(FWD)
Gi1/0/2
Cost 4
(FWD)
Gi1/0/2
Cost 4
(BLK)
Gi1/0/1
Cost 4
(FWD)
Gi1/0/3
Cost 4
(FWD)
Gi1/0/3
Cost 4
(FWD)
Gi1/0/1
Cost 4
(BLK)
Gi1/0/2
Cost 3
(FWD)
Cost lowered

You can investigate the STP port roles on SW1 and SW3 by using the **show spanning-tree** command, as shown in Example 28-5. Here you can see that interface GigabitEthernet1/0/2 now has a lower cost, and it is assigned as the root port (unlike in its original state). STP reconsiders the new lower-cost path between SW3 and SW2, and new port roles are assigned on SW1 and SW3. Because SW2 is the root bridge, it has all ports as designated (forwarding). Because SW3 has a lower-cost path to the root bridge (SW2), SW3 becomes the designated bridge for the link between SW1 and SW3.

Example 28-5 Verifying STP Port Cost and Port State

```
SW1# show spanning-tree
<... output omitted ...>
Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Gi1/0/1             Root  FWD 4         128.1    P2p
Gi1/0/2             Altn BLK 4         128.2    P2p

SW3# show spanning-tree
<... output omitted ...>
Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Gi1/0/1             Altn BLK 4         128.2    P2p
Gi1/0/2             Root FWD 3         128.3    P2p
Gi1/0/3             Desg FWD 4         128.4    P2p
```

Enabling and Verifying RSTP

You can use the **spanning-tree mode rapid-pvst** global configuration command to enable the Cisco Rapid-PVST+ version of STP on all switches. Use the **show spanning-tree** command to verify that RSTP is successfully enabled, as shown in Example 28-6. If all but one switch in the network is running RSTP, the interfaces that lead to legacy STP switches automatically fall back to PVST+. Port roles, port status, cost, and port ID remain as they were in Figure 28-15, but the network converges more quickly when RSTP is enabled.

Example 28-6 Configuring RSTP and Verifying STP Mode

```
SW1(config)# spanning-tree mode rapid-pvst

SW2(config)# spanning-tree mode rapid-pvst

SW3(config)# spanning-tree mode rapid-pvst
```

```
SW1# show spanning-tree

VLAN0001
  Spanning tree enabled protocol rstp
  <... output omitted ...>

SW2# show spanning-tree

VLAN0001
  Spanning tree enabled protocol rstp
  <... output omitted ...>
```

```
SW3# show spanning-tree

VLAN0001
  Spanning tree enabled protocol rstp
  <... output omitted ...>
```

STP Stability Mechanisms

Achieving and maintaining a loop-free STP topology revolves around the simple process of sending and receiving BPDUs. Under normal conditions, the loop-free topology is determined dynamically. This section reviews the STP features that can protect a network against unexpected BPDUs being received or the sudden loss of BPDUs. This section focuses on the following mechanisms:

- STP PortFast and BPDU Guard
- Root Guard
- Loop Guard
- Unidirectional Link Detection

STP PortFast and BPDU Guard

As previously discussed, if a switch port connects to another switch, the STP initialization cycle must transition from state to state to ensure a loop-free topology. However, for access devices such as PCs, laptops, servers, and printers, the delays that are incurred with STP initialization can cause problems such as DHCP timeouts. Cisco designed PortFast to reduce the time required for an access device to enter the forwarding state. STP is designed to prevent loops. Because there can be no loop on a port that is connected directly to a host or server, the full function of STP is not needed for that port. PortFast is a Cisco enhancement to STP that allows a switch port to begin forwarding much faster than a switch port in normal STP mode.

In a valid PortFast configuration, configuration BPDUs should never be received because access devices do not generate BPDUs. A BPDU that a port receives would indicate that another bridge or switch is connected to the port. This event could happen if a user plugged a switch on their desk into the port where the user PC was already plugged in.

The STP PortFast BPDU Guard enhancement allows network designers to enforce the STP domain borders and keep the active topology predictable. The devices behind the ports that have STP PortFast enabled are not able to influence the STP topology. At the reception of BPDUs, the BPDU Guard operation disables the port that has PortFast configured. The BPDU Guard mechanism transitions the port into *errdisable* state, and a message appears at the console. For example, the following message might appear:

```
%SPANTREE-2-BLOCK_BPDUGUARD: Received BPDU on port GigabitEthernet1/0/8 with BPDU
  guard enabled. Disabling port.
%PM-4-ERR_DISABLE: bpduguard error detected on Gi1/0/8, putting Gi1/0/8 in
  err-disable state
```

NOTE: Because the purpose of PortFast is to minimize the time that access ports that are connecting to user equipment and servers must wait for spanning tree to converge, you should use it only on access ports. If you enable PortFast on a port that is connecting to another switch, you risk creating a spanning-tree loop. Keep in mind that the BPDU Filter feature is available but not recommended. You should enable BPDU Guard on all PortFast-enabled ports to prevent a switch from being connected to a switch port that is dedicated for an end device.

The **spanning-tree bpduguard enable** interface configuration command configures BPDU Guard on an interface. The **spanning-tree portfast bpduguard default** global configuration command enables BPDU Guard globally for all PortFast-enabled ports.

The **spanning-tree portfast** interface configuration command configures PortFast on an interface. The **spanning-tree portfast default** global configuration command enables PortFast on all nontrunking interfaces.

Example 28-7 shows how to configure and verify PortFast and BPDU Guard on an interface on SW1 and globally on SW2.

Example 28-7 Configuring and Verifying PortFast and BPDU Guard

```
SW1(config)# interface GigabitEthernet 1/0/8
SW1(config-if)# spanning-tree portfast
SW1(config-if)# spanning-tree bpduguard enable

SW2(config)# spanning-tree portfast default
SW2(config)# spanning-tree portfast bpduguard default
```

```
SW1# show running-config interface GigabitEthernet1/0/8
<... output omitted ...>
interface GigabitEthernet1/0/8
<... output omitted ...>
spanning-tree portfast
spanning-tree bpduguard enable
end

SW2# show spanning-tree summary
<... output omitted ...>
Portfast Default             is enabled
PortFast BPDU Guard Default  is enabled
<... output omitted ...>

SW1# show spanning-tree interface GigabitEthernet1/0/8 portfast
VLAN0010             enabled
```

Note that the syntax for enabling PortFast can vary between switch models and IOS versions. For example, NX-OS uses the **spanning-tree port type edge** command to enable the PortFast feature. Since Cisco IOS Release 15.2(4)E or IOS XE 3.8.0E, if you enter the **spanning-tree portfast** command in the global or interface configuration mode, the system automatically saves it as **spanning-tree portfast edge**.

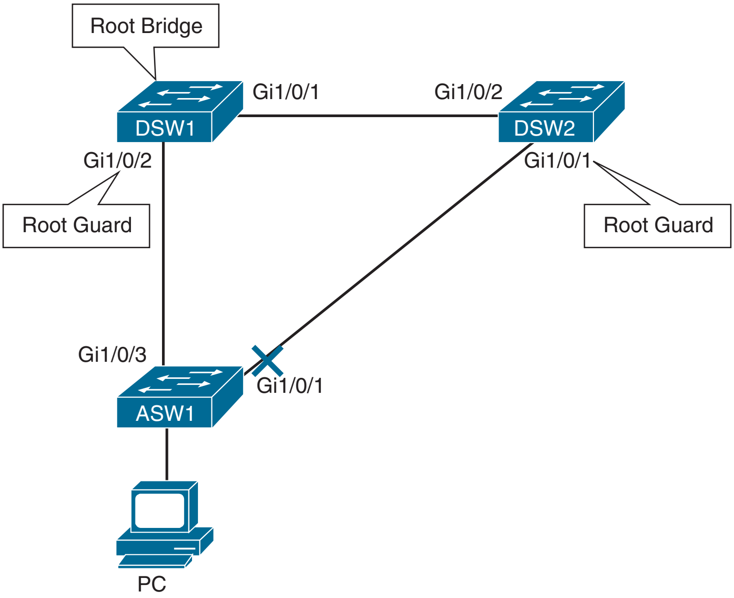
Root Guard

The Root Guard feature was developed to control where candidate root bridges can be connected and found on a network. Once a switch learns the current root bridge's bridge ID, if another switch advertises a superior BPDU, or one with a better bridge ID, on a port where Root Guard is enabled, the local switch does not allow the new switch to become the root. As long as the superior BPDUs are being received on the port, the port is kept in the *root-inconsistent* STP state. No data can be sent or received in that state, but the switch can listen to BPDUs received on the port to detect a new root advertising itself.

Use Root Guard on switch ports where you never expect to find the root bridge for a VLAN. When a superior BPDU is heard on the port, the entire port, in effect, becomes blocked.

In Figure 28-16, switches DSW1 and DSW2 are the core of the network. DSW1 is the root bridge for VLAN 1. ASW is an access layer switch. The link between DSW2 and ASW is blocking on the ASW side. ASW should never become the root bridge, so Root Guard is configured on DSW1 GigabitEthernet 1/0/2 and DSW2 GigabitEthernet 1/0/1. Example 28-8 shows the configuration of the Root Guard feature for the topology in Figure 28-16.

Figure 28-16 Root Guard Topology Example



Example 28-8 Configuring Root Guard

```
DSW1(config)# interface GigabitEthernet 1/0/2
DSW1(config-if)# spanning-tree guard root
%SPANTREE-2-ROOTGUARD_CONFIG_CHANGE: Root guard enabled on port
  GigabitEthernet1/0/2.

DSW2(config)# interface GigabitEthernet 1/0/1
DSW2(config-if)# spanning-tree guard root
%SPANTREE-2-ROOTGUARD_CONFIG_CHANGE: Root guard enabled on port
  GigabitEthernet1/0/1.
```

If a superior BPDU is received on a Root Guard port, the following message is sent to the console:

```
%SPANTREE-2-ROOTGUARD_BLOCK: Root guard blocking port GigabitEthernet1/0/2 on
  VLAN0001.
```

STP Loop Guard

The STP Loop Guard feature provides additional protection against Layer 2 loops. A Layer 2 loop is created when an STP blocking port in a redundant topology erroneously transitions to the forwarding state. This usually happens because one of the ports of a physically redundant topology (not necessarily the STP blocking port) no longer receives STP BPDUs. In its operation, STP relies on continuous reception or transmission of BPDUs based on the port role. The designated port transmits BPDUs, and the non-designated port receives BPDUs.

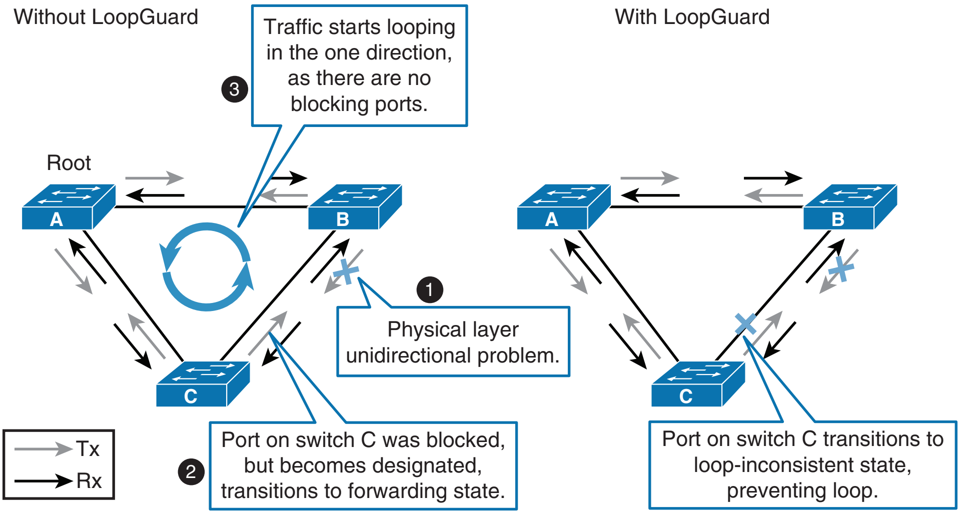
When one of the ports in a physically redundant topology no longer receives BPDUs, STP conceives that the topology is loop free. Eventually, the blocking port from the alternate or backup port becomes designated and moves to a forwarding state. This situation creates a loop, as shown in Figure 28-17.

The Loop Guard feature makes additional checks. If BPDUs are not received on a non-designated port, and Loop Guard is enabled, that port is moved into the STP *loop-inconsistent* blocking state instead of the listening/learning/forwarding state.

Once the BPDU is received on a port in a *loop-inconsistent* STP state, the port transitions into another STP state. According to the received BPDU, this means that the recovery is automatic, and intervention is not necessary.

Example 28-9 shows the configuration and verification of Loop Guard on switches SW1 and SW2. Notice that Loop Guard is configured at the interface level on SW1 and globally on SW2.

Figure 28-17 Loop Guard Example


Without LoopGuard
With LoopGuard
Traffic starts looping in the one direction, as there are no blocking ports.
3
Root
A
B
C
1
Physical layer unidirectional problem.
Tx
Rx
2
Port on switch C was blocked, but becomes designated, transitions to forwarding state.
A
B
C
Port on switch C transitions to loop-inconsistent state, preventing loop.

Example 28-9 Configuring and Verifying Loop Guard

```
SW1(config)# interface GigabitEthernet1/0/1
SW1(config-if)# spanning-tree guard loop

SW2(config)# spanning-tree loopguard default
```

```
SW1# show spanning-tree interface GigabitEthernet 1/0/1 detail
<...output omitted...>
   Loop guard is enabled on the port
   BPDU: send 6732, received 2846

SW2# show spanning-tree summary
Switch is in rapid-pvst mode
Root bridge for: none
Extended system ID                is enabled
Portfast Default                  is disabled
PortFast BPDU Guard Default       is disabled
Portfast BPDU Filter Default      is disabled
Loopguard Default                 is enabled
EtherChannel misconfig guard      is enabled
<...output omitted...>
```

Unidirectional Link Detection

Unidirectional Link Detection (UDLD) is a Cisco-proprietary protocol that detects unidirectional links and prevents Layer 2 loops from occurring across fiber-optic cables. UDLD is a Layer 2 protocol that works with the Layer 1 mechanisms to determine the physical status of a link. If one fiber strand in a pair is disconnected, autonegotiation prevents the link from becoming active or staying up. If both fiber strands are functional from a Layer 1 perspective, UDLD determines whether traffic is flowing bidirectionally between the correct neighbors.

The switch periodically transmits UDLD packets on an interface with UDLD enabled. If the packets are not echoed back within a specific time frame, the link is flagged as unidirectional, and the interface is *error disabled*. Devices on both ends of the link must support UDLD for the protocol to successfully identify and disable unidirectional links.

After UDLD detects a unidirectional link, it can take two courses of action, depending on the configured mode:

- **Normal mode:** In this mode, when a unidirectional link is detected, the port is allowed to continue its operation. UDLD just marks the port as having an undetermined state. A syslog message is generated.
- **Aggressive mode:** In this mode, when a unidirectional link is detected, the switch tries to reestablish the link. It sends one message per second, for 8 seconds. If none of these messages are sent back, the port is placed in an error-disabled state.

You configure UDLD on a per-port basis, although you can enable it globally for all fiber-optic switch ports (either native fiber or fiber-based GBIC or SFP modules). By default, UDLD is disabled on all switch ports. To enable it globally, use the global configuration command **udld** {**enable** | **aggressive** | **message time** *seconds*}.

For normal mode, use the **enable** keyword; for aggressive mode, use the **aggressive** keyword. You can use the **message time** keywords to set the message interval, in seconds, ranging from 1 to 90 seconds. The default interval is 15 seconds.

You also can enable or disable UDLD on individual switch ports, if needed, by using the interface configuration command **udld** {**enable** | **aggressive** | **disable**}.

You can use the **disable** keyword to completely disable UDLD on a fiber-optic interface.

Example 28-10 shows the configuration and verification of UDLD on SW1. Assume that UDLD is also enabled on its neighbor SW2.

Example 28-10 Configuring and Verifying UDLD

```
SW1(config)# udld aggressive

SW1# show udld GigabitEthernet2/0/1
Interface Gi2/0/1
---
Port enable administrative configuration setting: Enabled / in aggressive mode
Port enable operational state: Enabled / in aggressive mode
Current bidirectional state: Bidirectional
Current operational state: Advertisement - Single Neighbor detected
Message interval: 15000 ms
Time out interval: 5000 ms

<...output omitted...>

    Entry 1
    ---
    Expiration time: 37500 ms
    Cache Device Index: 1
    Current neighbor state: Bidirectional
    Device ID: 94DE32491I
    Port ID: Gi2/0/1
    Neighbor echo 1 device: 9M34622MQ2
    Neighbor echo 1 port: Gi2/0/1

    TLV Message interval: 15 sec
    No TLV fast-hello interval
    TLV Time out interval: 5
    TLV CDP Device name: SW2

SW1# show udld neighbors
Port     Device Name          Device ID  Port ID  Neighbor State
-------- -------------------- ---------- -------- --------------
Gi2/0/1  SW1                  1          Gi2/0/1  Bidirectional
```

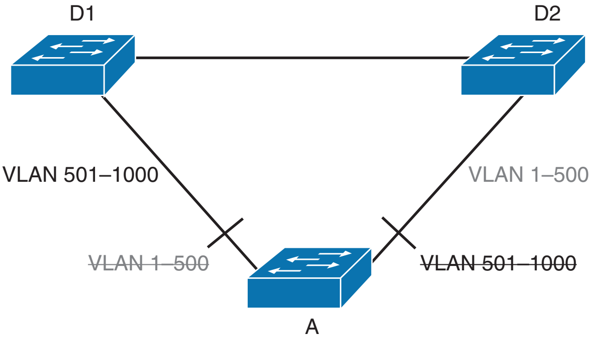
Multiple Spanning Tree Protocol

The main purpose of Multiple Spanning Tree Protocol (MST) is to reduce the total number of spanning-tree instances to match the physical topology of the network. Reducing the total number of spanning-tree instances reduces the CPU loading of a switch. The number of instances of spanning tree is reduced to the number of links (that is, active paths) that are available.

In a scenario where PVST+ is implemented, there could be up to 4094 instances of spanning tree, each with its own BPDU conversations, root bridge elections, and path selections.

Figure 28-18 illustrates an example where the goal would be to achieve load distribution with VLANs 1 through 500 using one path and VLANs 501 through 1000 using the other path. Instead of creating 1000 PVST+ instances, you can use MST with only two instances of spanning tree. The two ranges of VLANs are mapped to two MST instances, respectively. Rather than maintain 1000 spanning trees, each switch needs to maintain only two.

Figure 28-18 VLAN Load Balancing Example



Implemented in this fashion, MST converges faster than PVST+ and is backward compatible with 802.1D STP, 802.1w RSTP, and the Cisco PVST+ architecture. Implementation of MST is not required if the Cisco enterprise campus architecture is being employed because the number of active VLAN instances, and hence the number of STP instances, would be small and very stable due to the design.

MST allows you to build multiple spanning trees over trunks by grouping VLANs and associating them with spanning-tree instances. Each instance can have a topology independent of other spanning-tree instances. This architecture provides multiple active forwarding paths for data traffic and enables load balancing.

With MST, network fault tolerance is improved over CST (Common Spanning Tree) because a failure in one instance (forwarding path) does not necessarily affect other instances. The VLAN-to-MST grouping must be consistent across all bridges within an MST region. Interconnected bridges that have the same MST configuration are referred to as an *MST region*.

You must configure a set of bridges with the same MST configuration information to enable them to participate in a specific set of spanning-tree instances. Bridges with different MST configurations or legacy bridges running 802.1D are considered separate MST regions. MST is defined in the IEEE 802.1s standard and has been part of the 802.1Q standard since 2005.

MST Regions

MST differs from the other spanning-tree implementations in that it combines some, but not necessarily all, VLANs into logical spanning-tree instances. With MST, there is a problem of determining which VLAN is to be associated with which instance. More precisely, this issue means tagging BPDUs so that receiving devices can identify the instances and the VLANs to which they apply.

The issue is irrelevant in the case of the 802.1D standard, in which all instances are mapped to a unique CST instance. In the PVST+ implementation, different VLANs carry the BPDUs for their respective instances (one BPDU per VLAN), based on the VLAN tagging information. To provide this logical assignment of VLANs to spanning trees, each switch that is running MST in the network has a single MST configuration consisting of three attributes:

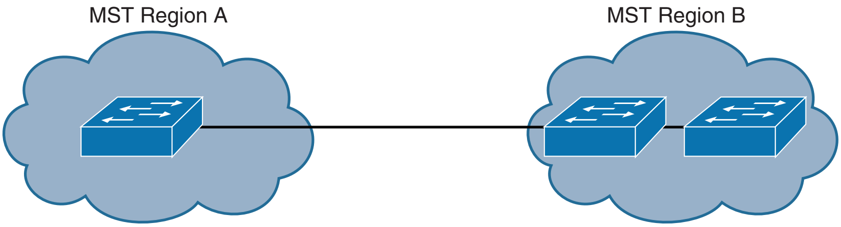
- An alphanumeric configuration name (32 bytes)
- A configuration revision number (2 bytes)
- A table that associates each potential VLAN supported on the chassis with a given instance

To ensure a consistent VLAN-to-instance mapping, it is necessary for the protocol to be able to identify the boundaries of the regions exactly. For that purpose, the characteristics of the region are included in BPDUs. The exact VLAN-to-instance mapping is not propagated in the BPDU because the switches need to know only whether they are in the same region as a neighbor.

Therefore, only a digest of the VLAN-to-instance-mapping table is sent, along with the revision number and the name. After a switch receives a BPDU, it extracts the digest (a numeric value that is derived from the VLAN-to-instance-mapping table through a mathematical function) and compares it with its own computed digest. If the digests differ, the mapping must be different, so the port on which the BPDU was received is at the boundary of a region.

In generic terms, a port is at the boundary of a region if the designated bridge on its segment is in a different region or if it receives legacy 802.1D BPDUs. Figure 28-19 illustrates the concept of MST regions and boundary ports.

Figure 28-19 MST Regions



The configuration revision number gives you a method of tracking the changes that are made to an MST region. This number does not automatically increase each time that you make changes to the MST configuration. Each time you make a change, you should increase the revision number by one.

MST Instances

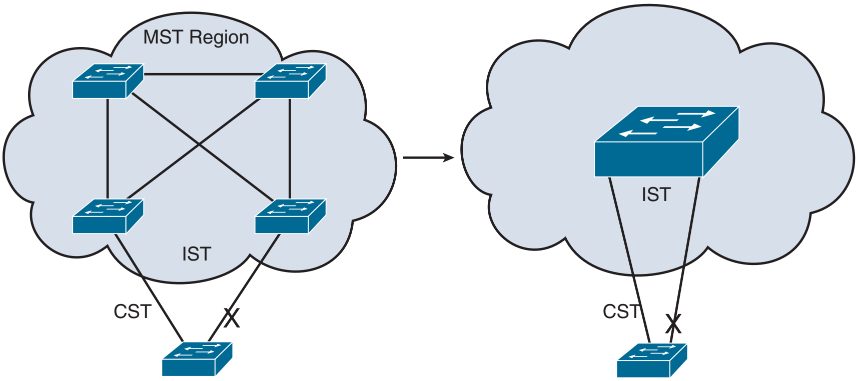
MST was designed to interoperate with all other forms of STP. Therefore, it also must support STP instances from each STP type. This is where MST can get confusing. Think of the entire enterprise network as having a single CST topology so that one instance of STP represents any and all VLANs and MST regions present. CST maintains a common loop-free topology and integrates all forms of STP that might be in use. To do this, CST must regard each MST region as a single "black box" bridge because it has no idea what is inside the region (and it does not care). CST maintains a loop-free topology only with the links that connect the regions to each other and to standalone switches running 802.1Q CST.

Something other than CST must work out a loop-free topology inside each MST region. Within a single MST region, an Internal Spanning Tree (IST) instance runs to work out a loop-free topology between the links where CST meets the region boundary and all switches inside the region. Think of the IST instance as a locally significant CST, bounded by the edges of the region.

IST presents the entire region as a single virtual bridge to CST outside. BPDUs are exchanged at the region boundary only over the native VLAN of trunks.

Figure 28-20 shows the basic concept behind the IST instance. The network at the left has an MST region, where several switches are running compatible MST configurations. Another switch is outside the region because it is running only CST from 802.1Q.

Figure 28-20 MST, IST, and CST Example

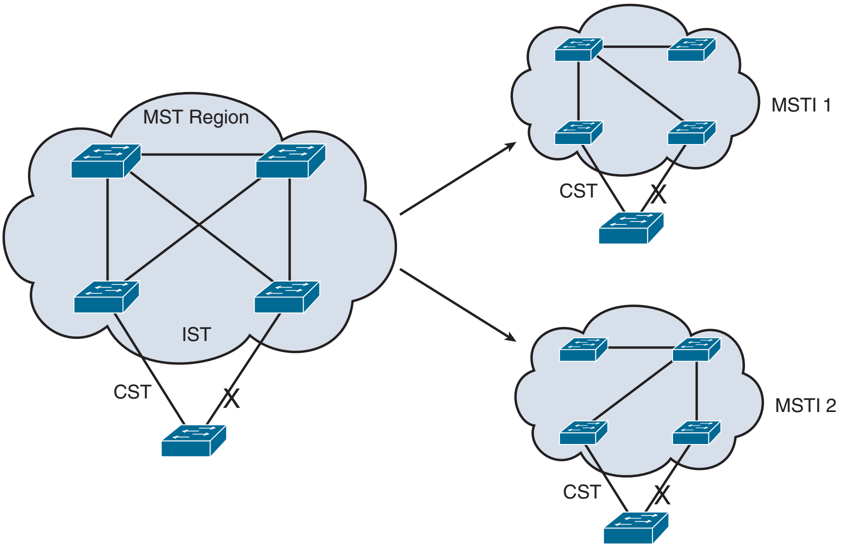


The same network is shown at the right, where IST has produced a loop-free topology for the network inside the region. IST makes the internal network look like a single bridge (the "big switch" in the cloud) that can interface with CST running outside the region.

Recall that the whole idea behind MST is the capability to map multiple VLANs to a smaller number of STP instances. Inside a region, the actual MST instances (MSTI) exist alongside IST. Cisco supports a maximum of 16 MST instances in each region. IST always exists as MST instance number 0, leaving MST instances 1 through 15 available for use.

Figure 28-21 shows how different MST instances can exist within a single MST region. The left portion of the figure is identical to that of Figure 28-20. In this network, two MST instances, MSTI 1 and MSTI 2, are configured with different VLANs mapped to each. Their topologies follow the same structure as the network on the left side of the figure, but they have converged differently.

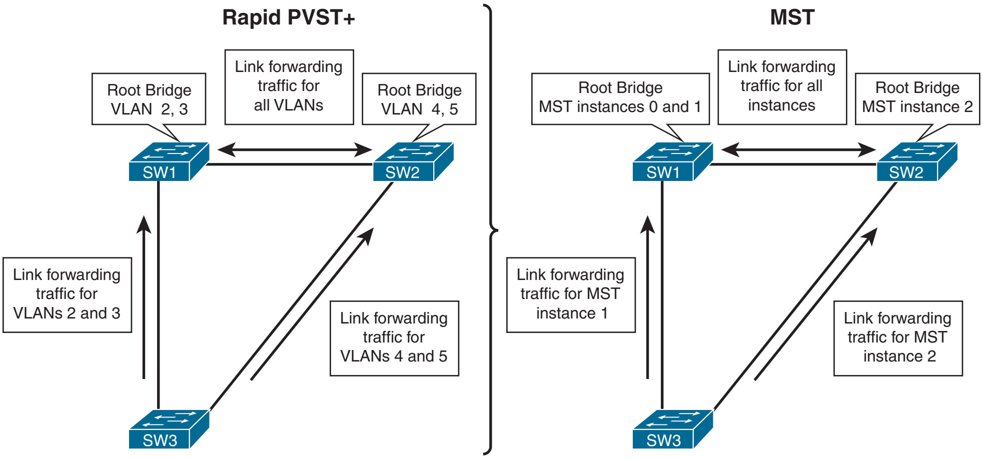
Figure 28-21 MST Instances



MST Configuration and Verification

The left side of Figure 28-22 shows an initial STP configuration. All three switches are configured with Rapid PVST+ and four user-created VLANs: 2, 3, 4, and 5. SW1 is configured as the root bridge for VLANs 2 and 3. SW2 is configured as the root bridge for VLANs 4 and 5. This configuration distributes forwarding of traffic between the SW3–SW1 and SW3–SW2 uplinks.

Figure 28-22 MST Configuration Topology


Rapid PVST+
Root Bridge VLAN 2, 3
Link forwarding traffic for all VLANs
Root Bridge VLAN 4, 5
SW1
SW2
Link forwarding traffic for VLANs 2 and 3
Link forwarding traffic for VLANs 4 and 5
SW3
MST
Root Bridge MST instances 0 and 1
Link forwarding traffic for all instances
Root Bridge MST instance 2
SW1
SW2
Link forwarding traffic for MST instance 1
Link forwarding traffic for MST instance 2
SW3

The right side of Figure 28-22 shows the STP configuration after VLANs 2 and 3 are mapped into MST Instance 1 and VLANs 4 and 5 are mapped into MST Instance 2.

Example 28-11 shows the commands to configure and verify MST on all three switches in order to achieve the desired load balancing shown in Figure 28-22.

Example 28-11 Configuring MST

```
SW1(config)# spanning-tree mode mst
SW1(config)# spanning-tree mst 0 root primary
SW1(config)# spanning-tree mst 1 root primary
SW1(config)# spanning-tree mst 2 root secondary
SW1(config)# spanning-tree mst configuration
SW1(config-mst)# name 31DAYS
SW1(config-mst)# revision 1
SW1(config-mst)# instance 1 vlan 2,3
SW1(config-mst)# instance 2 vlan 4,5

SW2(config)# spanning-tree mode mst
SW2(config)# spanning-tree mst 0 root secondary
SW2(config)# spanning-tree mst 1 root secondary
SW2(config)# spanning-tree mst 2 root primary
SW2(config)# spanning-tree mst configuration
```

```
SW2(config-mst)# name 31DAYS
SW2(config-mst)# revision 1
SW2(config-mst)# instance 1 vlan 2,3
SW2(config-mst)# instance 2 vlan 4,5

SW3(config)# spanning-tree mode mst
SW3(config)# spanning-tree mst configuration
SW3(config-mst)# name 31DAYS
SW3(config-mst)# revision 1
SW3(config-mst)# instance 1 vlan 2,3
SW3(config-mst)# instance 2 vlan 4,5
```

In the configuration shown in Example 28-11, SW1 is configured as the primary root bridge for Instances 0 and 1, and SW2 is configured as the primary root for Instance 2. The three switches are configured with identical region names, revision numbers, and VLAN instance mappings.

Example 28-12 shows the commands for verifying MST. Figure 28-23 shows the interfaces referenced in this output.

Example 28-12 Verifying MST

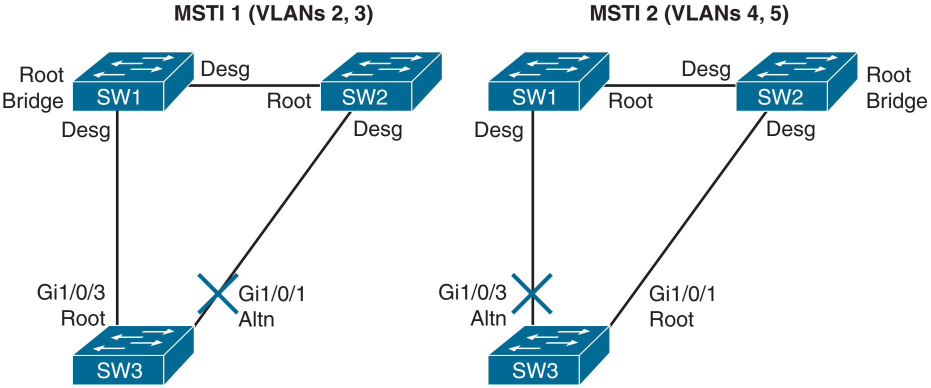
```
SW3# show spanning-tree mst configuration
Name       [31DAYS]
Revision  1     Instances configured 3

Instance   Vlans mapped
--------   -----------------------------------------------------------
0          1,6-4094
1          2-3
2          4-5
```

```
SW3# show spanning-tree mst 1

##### MST1    vlans mapped:    2-3
<... output omitted ..>
Gi1/0/1              Altn BLK 20000     128.1     P2p
Gi1/0/3              Root FWD 20000     128.3     P2p
<... output omitted ..>

SW3# show spanning-tree mst 2

##### MST2    vlans mapped:    4-5
<... output omitted ..>
Gi1/0/1              Root FWD 20000     128.1     P2p
Gi1/0/3              Altn BLK 20000     128.3     P2p
<... output omitted ..>
```

VLANs 2 and 3 are mapped to MSTI 1. VLANs 4 and 5 are mapped to MSTI 2. All other VLANs are mapped to MSTI 0 or IST.

MST Instances 1 and 2 have two distinct Layer 2 topologies. Instance 1 uses the uplink toward SW1 as the active link and blocks the uplink toward SW2. Instance 2 uses the uplink toward SW2 as the active link and blocks uplink toward SW1, as shown in Figure 28-23.

Figure 28-23 MST Configuration Topology



Configuring MST Path Cost and Port Priority

You can assign lower-cost values to interfaces that you want selected first and higher-cost values to interfaces that you want selected last. If all interfaces have the same cost value, MST puts the interface with the lowest sender port ID in the forwarding state and blocks the other interfaces.

To change the STP cost of an interface, enter interface configuration mode for that interface and use the command **spanning-tree mst** *instance* **cost** *cost*. For the *instance* variable, you can specify a single instance, a range of instances that are separated by a hyphen, or a series of instances that are separated by a comma. The range is 0 to 4094. For the *cost* variable, the range is 1 to 200000000; the default value is usually derived from the media speed of the interface.

You can assign higher sender priority values (lower numeric values) to interfaces that you want selected first and lower sender priority values (higher numeric values) to interfaces that you want selected last. If all sender interfaces have the same priority value, MST puts the interface with the lowest sender port ID in the forwarding state and blocks the other interfaces.

To change the STP port priority of an interface, enter interface configuration mode and use the **spanning-tree mst** *instance* **port-priority** *priority* command. For the *priority* variable, the range is 0 to 240, in increments of 16. The default is 128. The lower the number, the higher the priority.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	**Module, Chapter, or Link**
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	2, 3, 4
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	2

Day 27

Port Aggregation

ENCOR 350-401 Exam Topics

Infrastructure

- Layer 2
  - Troubleshoot static and dynamic EtherChannels

Key Topics

Today we review configuring, verifying, and troubleshooting Layer 2 and Layer 3 EtherChannels. EtherChannel is a port link aggregation technology that allows multiple physical port links to be grouped into one single logical link. It is used to provide high-speed links and redundancy in a campus network and data centers. Today we also review the two EtherChannel protocols supported on Cisco Catalyst switches: Cisco's proprietary Port Aggregation Protocol (PAgP) and the IEEE standard Link Aggregation Control Protocol (LACP). LACP was initially standardized as 802.3ad but was formally transferred to the 802.1 group in 2008, with the publication of IEEE 802.1AX.

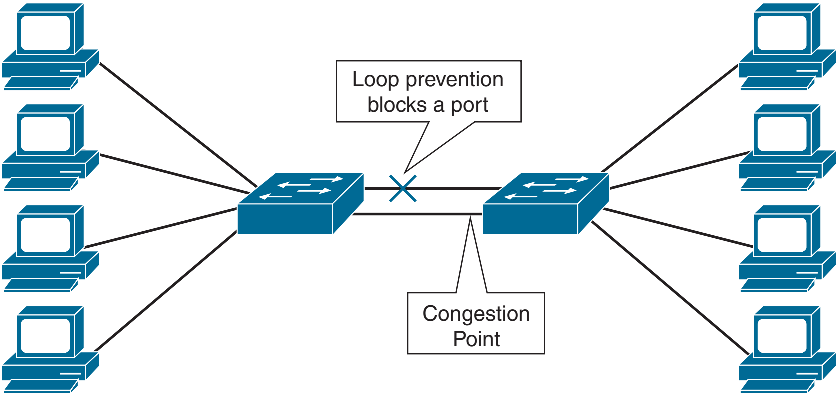
Need for EtherChannel

EtherChannel allows multiple physical Ethernet links to be combined into one logical channel. This process allows load sharing of traffic among the links in the channel and redundancy in case one or more links in the channel fail. EtherChannel can be used to interconnect LAN switches, routers, and servers.

The proliferation of bandwidth-intensive applications such as video streaming and cloud-based storage has caused a need for greater network speeds and scalable bandwidth. You can increase network speed by using faster links, but faster links are more expensive. Furthermore, such a solution cannot scale indefinitely and experiences a limitation where the fastest possible port is no longer fast enough.

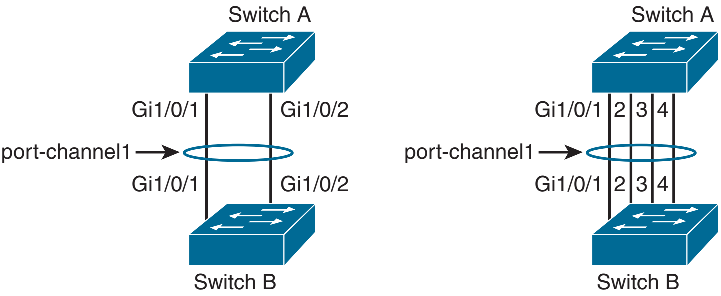
You can also increase network speeds by using more physical links between switches. When multiple links aggregate on a switch, congestion can occur. One solution is to increase uplink speed, but that solution cannot scale indefinitely. Another solution is to multiply uplinks, but loop-prevention mechanisms such as Spanning Tree Protocol (STP) disable some ports. Figure 27-1 shows that simply adding an extra link between switches doesn't increase the bandwidth available between both devices because STP blocks one of the links.

Figure 27-1 Multiple Links with STP



EtherChannel technology provides a solution. EtherChannel was originally developed by Cisco as a means of increasing speed between switches by grouping several Fast Ethernet or Gigabit Ethernet ports into one logical EtherChannel link collectively known as a *port channel*, as shown in Figure 27-2. Because the two physical links are bundled into a single EtherChannel, STP no longer sees two physical links. Instead, it sees a single EtherChannel. As a result, STP does not need to block one of the physical links to prevent a loop. Because all physical links in the EtherChannel are active, bandwidth is increased. EtherChannel provides the additional bandwidth without requiring you to upgrade links to a faster and more expensive connection because it relies on existing switch ports. Figure 27-2 also shows an example of four physical links being bundled into one logical port channel.

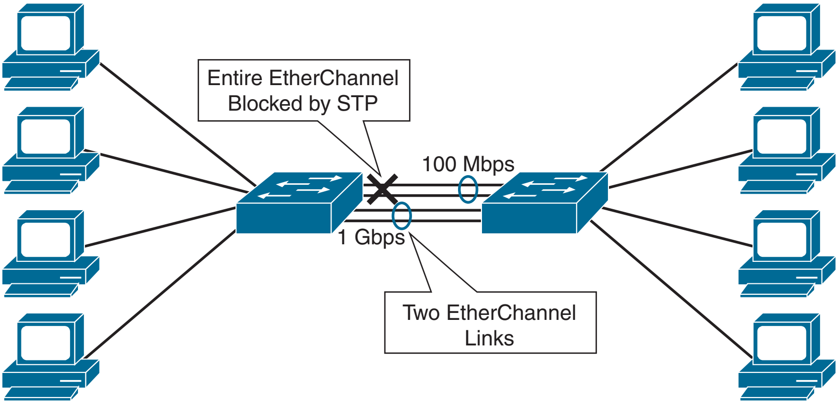
Figure 27-2 Scaling Bandwidth by Bundling Physical Links into an EtherChannel



You can group from 2 to 8 (or 16, on some newer models) physical ports into a logical EtherChannel link, but you cannot mix port types within a single EtherChannel. For example, you could group 4 Fast Ethernet ports into 1 logical Ethernet link, but you could not group 2 Fast Ethernet ports and 2 Gigabit Ethernet ports into 1 logical Ethernet link.

You can also configure multiple EtherChannel links between two devices. When several EtherChannels exist between two switches, STP may block one of the EtherChannels to prevent redundant links. When STP blocks one of the redundant links, it blocks an entire EtherChannel, thus blocking all the ports belonging to that EtherChannel link, as shown in Figure 27-3.

Figure 27-3 Multiple EtherChannel Links and STP



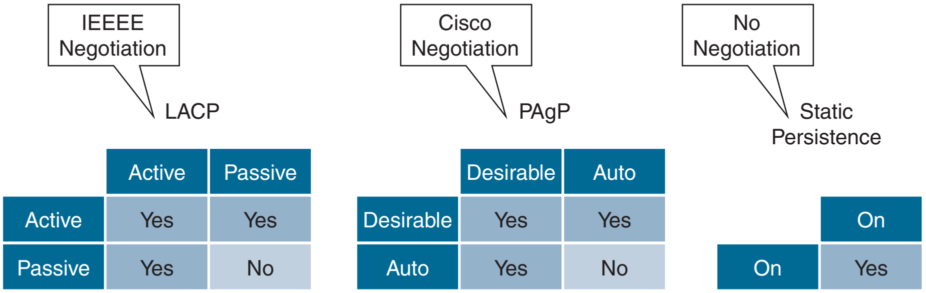
In addition to providing higher bandwidth, EtherChannel provides several other advantages:

- You can perform most configuration tasks on the EtherChannel interface instead of on each individual port, which ensures configuration consistency throughout the links.
- Because EtherChannel relies on the existing switch ports, you do not need to upgrade the link to a faster and more expensive connection to obtain more bandwidth.
- Load balancing is possible between links that are part of the same EtherChannel. Depending on your hardware platform, you can implement one or several load-balancing methods, such as source MAC-to-destination MAC or source IP-to-destination IP load balancing, across the physical links.
- EtherChannel creates an aggregation that is seen as one logical link. When several EtherChannel bundles exist between two switches, STP may block one of the bundles to prevent redundant links. When STP blocks one of the redundant links, it blocks one EtherChannel, thus blocking all the ports belonging to that EtherChannel link. Where there is only one EtherChannel link, all physical links in the EtherChannel are active because STP sees only one (logical) link.
- EtherChannel provides redundancy. The loss of a physical link within an EtherChannel does not create a change in the topology, and you don't need a spanning-tree recalculation. If at least one physical link is active, the EtherChannel is functional, even if its overall throughput decreases.

EtherChannel Mode Interactions

EtherChannel can be established using one of three mechanisms: LACP, PAgP, or static persistence (see Figure 27-4).

Figure 27-4 EtherChannel Modes



	Active	Passive
Active	Yes	Yes
Passive	Yes	No

	Desirable	Auto
Desirable	Yes	Yes
Auto	Yes	No

	On
On	Yes

LACP

LACP enables several physical ports to be bundled together to form a single logical channel. LACP allows a switch to negotiate an automatic bundle by sending LACP packets to the peer using MAC address 0180.c200.0002. Because LACP is an IEEE standard, you can use it to facilitate EtherChannels in mixed-switch environments. LACP checks for configuration consistency and manages link additions and failures between two switches. It ensures that when EtherChannel is created, all ports have the same type of configuration speed, duplex setting, and VLAN information. Any port channel modification after the creation of the channel also changes all the other channel ports.

LACP control packets are exchanged between switches over EtherChannel-capable ports. Port capabilities are learned and compared with local switch capabilities. LACP assigns roles to the EtherChannel ports. The switch with the lowest system priority is allowed to make decisions about what ports actively participate in EtherChannel. Ports become active according to their port priority. A lower number means higher priority. Commonly, up to 16 links can be assigned to an EtherChannel, but only 8 can be active at a time. Nonactive links are placed into a hot standby state and are enabled if one of the active links goes down.

The maximum number of active links in an EtherChannel varies depending on the switch.

The LACP modes of operation are as follows:

- **Active:** Enable LACP unconditionally. The port sends LACP requests to connected ports.
- **Passive:** Enable LACP only if an LACP device is detected. The port waits for LACP requests and responds to requests for LACP negotiation.

Use the **channel-group** *channel-group-number* **mode** {**active** | **passive**} interface configuration command to enable LACP.

PAgP

PAgP provides the same negotiation benefits as LACP. PAgP is a Cisco-proprietary protocol that works only on Cisco devices. PAgP packets are exchanged between switches over EtherChannel-capable ports using MAC address 0100.0ccc.cccc. Neighbors are identified, and capabilities are learned and compared with local switch capabilities. Ports that have the same capabilities are bundled together into an EtherChannel. PAgP forms an EtherChannel only on ports that are configured for identical VLANs or trunking. For example, PAgP groups the ports with the same speed, duplex mode, native VLAN, VLAN range, and trunking status and type. After grouping the links into an EtherChannel, PAgP adds the group to the spanning tree as a single device port.

PAgP has several modes of operation:

- **Desirable:** This mode enables PAgP unconditionally. In other words, the port starts actively sending negotiation messages to other ports.
- **Auto:** This mode enables PAgP only if a PAgP device is detected. In other words, the port waits for requests and responds to requests for PAgP negotiation, which reduces the transmission of PAgP packets. Negotiation with either LACP or PAgP introduces overhead and delay in initialization.
- **Silent:** If a switch is connected to a partner that is PAgP-capable, you can configure the switch port for non-silent operation by using the **non-silent** keyword. If you do not specify non-silent with the auto or desirable mode, silent mode is assumed. Using non-silent mode results in faster establishment of the EtherChannel when connecting to another PAgP neighbor.

Use the **channel-group** *channel-group-number* **mode** {**auto** | **desirable**} [**non-silent**] interface configuration command to enable PAgP.

Static

EtherChannel static **on** mode can be used to manually configure an EtherChannel. The static **on** mode forces a port to join an EtherChannel without negotiation. The **on** mode can be useful when a remote device does not support PAgP or LACP. In the **on** mode, a usable EtherChannel exists only when the devices at both ends of the link are configured in the **on** mode.

Ports that are configured in the **on** mode in the same channel group must have compatible port characteristics, such as speed and duplex. Ports that are not compatible are suspended, even though they are configured in the **on** mode.

Use the **channel-group** *channel-group-number* **mode on** interface configuration command to enable static **on** mode.

EtherChannel Configuration Guidelines

If improperly configured, some EtherChannel ports are automatically disabled to avoid network loops and other problems. Follow these guidelines to avoid configuration problems:

- Configure all ports in an EtherChannel to operate at the same speed and duplex mode.
- Enable all ports in an EtherChannel. A port in an EtherChannel that is disabled by using the **shutdown** interface configuration command is treated as a link failure, and its traffic is transferred to one of the remaining ports in the EtherChannel.
- When a group is first created, all ports follow the parameters set for the first port added to the group. If you change the configuration of one of these parameters, you must also make the changes to all ports in the group:
    - Allowed-VLAN list
    - Spanning-tree path cost for each VLAN

  - Spanning-tree port priority for each VLAN
  - Spanning-tree PortFast setting
- Assign all ports in the EtherChannel to the same VLAN or configure them as trunks. Ports with different native VLANs cannot form an EtherChannel.
- An EtherChannel supports the same allowed range of VLANs on all the ports in a trunking Layer 2 EtherChannel. If the allowed range of VLANs is not the same, the ports do not form an EtherChannel, even when PAgP is set to the auto or desirable mode.
- Ports with different spanning-tree path costs can form an EtherChannel if they are otherwise compatibly configured. Setting different spanning-tree path costs does not, by itself, make ports incompatible for the formation of an EtherChannel.
- For Layer 3 EtherChannel, because the port channel interface is a routed port, the **no switchport** command is applied to it. The physical interfaces are, by default, switched, which is a mode that is incompatible with a router port. The **no switchport** command is applied also to the physical ports to make their mode compatible with the EtherChannel interface mode.
- For Layer 3 EtherChannels, assign the Layer 3 address to the port channel logical interface, not to the physical ports in the channel.

EtherChannel Load Balancing Options

EtherChannel performs load balancing of traffic across links in a bundle. However, traffic is not necessarily distributed equally between all the links. Table 27-1 shows some of the possible hashing algorithms available.

Table 27-1 Types of EtherChannel Load Balancing Methods

Load Balancing Method	Hash Input Description
dst-ip	Destination IP address
dst-mac	Destination MAC address
dst-mixed-ip-port	Destination IP address and TCP/UDP port
src-dst-ip	Source and destination IP address
src-dst-mac	Source and destination MAC address
src-dst-mixed-ip-port	Source and destination IP address and TCP/UDP port
src-ip	Source IP address
src-mac	Source MAC address
src-mixed-ip-port	Source IP address and TCP/UDP port
src-port	Source port number
dst-port	Destination port number
src-dst-port	Source and destination port number

You can verify which load-balancing options are available on a device by using the **port-channel load-balance ?** global configuration command. (Remember that **?** shows all options for a command.)

The hashing algorithm calculates a binary pattern that selects a link within the EtherChannel bundle to forward the frame.

To achieve the optimal traffic distribution, always bundle an even number of links. For example, if you use four links, the algorithm looks at the last 2 bits, and 2 bits mean four indexes: 00, 01, 10, and 11. Each link in the bundle is assigned one of these indexes. If you bundle only three links, the algorithm still needs to use 2 bits to make decisions. One of the three links in the bundle will be utilized more than other two. With four links, the algorithm strives to load balance traffic in a 1:1:1:1 ratio. With three links, the algorithm strives to load balance traffic in a 2:1:1 ratio.

Use the **show etherchannel load-balance** command to verify how a switch will load balance network traffic, as illustrated in Example 27-1.

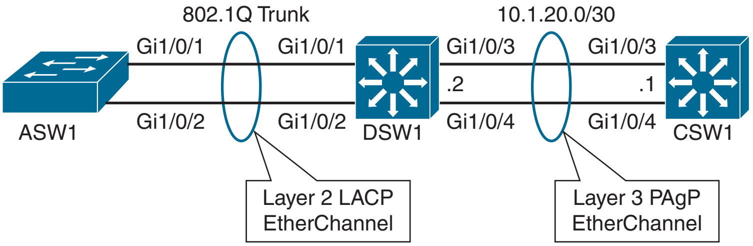
Example 27-1 Verifying EtherChannel Load Balancing

```
SW1# show etherchannel load-balance
EtherChannel Load-Balancing Configuration:
        src-dst-ip

EtherChannel Load-Balancing Addresses Used Per-Protocol:
Non-IP: Source XOR Destination MAC address
  IPv4: Source XOR Destination IP address
  IPv6: Source XOR Destination IP address
```

EtherChannel Configuration and Verification

This section shows how to configure and verify LACP and PAgP EtherChannels. Figure 27-5 illustrates the topology used in this section. Example 27-2 shows the commands used to configure a Layer 2 LACP EtherChannel trunk between ASW1 and DSW1, and Example 27-3 shows the commands used to configure a Layer 3 PAgP EtherChannel link between DSW1 and CSW1 using the 10.1.20.0/30 subnet.

Figure 27-5 EtherChannel Configuration Topology Example



Example 27-2 Configuring LACP Layer 2 EtherChannel

```
ASW1(config)# interface range GigabitEthernet 1/0/1-2
ASW1(config-if-range)# channel-group 1 mode passive
Creating a port-channel interface Port-channel 1
ASW1(config-if-range)# interface port-channel 1
ASW1(config-if)# switchport mode trunk
04:23:49.619: %LINEPROTO-5-UPDOWN: Line protocol on Interface
   GigabitEthernet1/0/1, changed state to down
04:23:49.628: %LINEPROTO-5-UPDOWN: Line protocol on Interface
   GigabitEthernet1/0/2, changed state to down
04:23:56.827: %EC-5-L3DONTBNDL2: Gi1/0/1 suspended: LACP currently not enabled on
   the remote port.
04:23:57.252: %EC-5-L3DONTBNDL2: Gi1/0/2 suspended: LACP currently not enabled on
   the remote port.
```

```
DSW1(config)# interface range GigabitEthernet 1/0/1-2
DSW1(config-if-range)# channel-group 1 mode active
Creating a port-channel interface Port-channel 1
DSW1(config-if-range)# interface port-channel 1
DSW1(config-if)# switchport mode trunk
04:25:39.823: %LINK-3-UPDOWN: Interface Port-channel1, changed state to up
04:25:39.869: %LINEPROTO-5-UPDOWN: Line protocol on Interface Port-channel1,
   changed state to up
```

Notice in Example 27-2 that ASW1 is configured as LACP passive and DSW1 is configured as LACP active. Also, because ASW1 is configured first, LACP suspends the bundled interfaces until DSW1 is configured. At that point the port channel state changes to up, and the link becomes active.

Example 27-3 Configuring PAgP Layer 3 EtherChannel

```
DSW1(config)# interface range GigabitEthernet 1/0/3-4
DSW1(config-if-range)# no switchport
05:27:24.765: %LINK-3-UPDOWN: Interface GigabitEthernet1/0/3, changed state to up
05:27:24.765: %LINK-3-UPDOWN: Interface GigabitEthernet1/0/4, changed state to up
05:27:25.774: %LINEPROTO-5-UPDOWN: Line protocol on Interface
   GigabitEthernet1/0/3, changed state to up
05:27:25.774: %LINEPROTO-5-UPDOWN: Line protocol on Interface
   GigabitEthernet1/0/4, changed state to up
DSW1(config-if-range)# channel-group 2 mode auto non-silent
Creating a port-channel interface Port-channel 2
05:29:08.169: %EC-5-L3DONTBNDL1: Gi1/0/3 suspended: PAgP not enabled on the
   remote port.
05:29:08.679: %EC-5-L3DONTBNDL1: Gi1/0/4 suspended: PAgP not enabled on the
   remote port.
```

```
DSW1(config-if-range)# interface port-channel 2
DSW1(config-if)# ip address 10.1.20.2 255.255.255.252
```

```
CSW1(config)# interface range GigabitEthernet 1/0/3-4
CSW1(config-if-range)# no switchport
05:32:16.839: %LINK-3-UPDOWN: Interface GigabitEthernet1/0/3, changed state to up
05:32:16.839: %LINK-3-UPDOWN: Interface GigabitEthernet1/0/4, changed state to up
05:32:17.844: %LINEPROTO-5-UPDOWN: Line protocol on Interface
  GigabitEthernet1/0/3, changed state to up
05:32:17.844: %LINEPROTO-5-UPDOWN: Line protocol on Interface
  GigabitEthernet1/0/4, changed state to up
CSW1(config-if-range)# channel-group 2 mode desirable non-silent
Creating a port-channel interface Port-channel 2
05:32:36.383: %LINEPROTO-5-UPDOWN: Line protocol on Interface Port-channel2,
  changed state to up
CSW1(config-if-range)# interface port-channel 2
CSW1(config-if)# ip address 10.1.20.1 255.255.255.252
```

In Example 27-3, DSW1 uses the PAgP auto non-silent mode, and CSW1 uses the PAgP desirable non-silent mode. Non-silent mode is used here because both switches are PAgP enabled. The **no switchport** command puts the physical interfaces into Layer 3 mode but notice that the actual IP address is configured on the port channel. The port channel inherited Layer 3 functionality when the physical interfaces were assigned to it.

To verify the state of the newly configured EtherChannels, you can use the following commands, as shown in Example 27-4:

- **show etherchannel summary**
- **show interfaces port-channel**
- **show lacp neighbor**
- **show pagp neighbor**

Example 27-4 Verifying EtherChannel

```
DSW1# show etherchannel summary
Flags:  D - down         P - bundled in port-channel
        I - stand-alone  s - suspended
        H - Hot-standby (LACP only)
        R - Layer3       S - Layer2
        U - in use       N - not in use, no aggregation
        f - failed to allocate aggregator

        M - not in use, minimum links not met
        m - not in use, port not aggregated due to minimum links not met
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port

        A - formed by Auto LAG


Number of channel-groups in use: 2
Number of aggregators:           2

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(SU)         LACP      Gi1/0/1(P)   Gi1/0/2(P)
2      Po2(RU)         PAgP      Gi1/0/3(P)   Gi1/0/4(P)
```

```
DSW1# show interfaces Port-channel 1
Port-channel1 is up, line protocol is up (connected)
  Hardware is EtherChannel, address is aabb.cc00.0130 (bia aabb.cc00.0130)
  MTU 1500 bytes, BW 2000000 Kbit/sec, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full-duplex, 1000Mb/s, link type is auto, media type is unknown
  input flow-control is off, output flow-control is unsupported
  Members in this channel: Gi1/0/1 Gi1/0/2
<. . . output omitted . . .>
```

```
DSW1# show lacp neighbor
Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode

Channel group 1 neighbors

                 LACP port                        Admin  Oper   Port     Port
Port       Flags   Priority  Dev ID          Age   key    Key    Number   State
Gi1/0/1    SA      32768     aabb.cc80.0300  20s   0x0    0x1    0x102    0x3C
Gi1/0/2    SA      32768     aabb.cc80.0300  23s   0x0    0x1    0x103    0x3C
```

```
DSW1# show pagp neighbor
Flags:  S - Device is sending Slow hello.  C - Device is in Consistent state.
        A - Device is in Auto mode.        P - Device learns on physical port.

Channel group 2 neighbors
          Partner               Partner          Partner         Partner  Group
Port      Name                  Device ID        Port       Age  Flags    Cap.
Gi1/0/3   CSW1                  aabb.cc80.0200   Gi1/0/3     6s  SC       20001
Gi1/0/4   CSW1                  aabb.cc80.0200   Gi1/0/4    16s  SC       20001
```

In the **show etherchannel summary** command output, you get confirmation that Port-Channel 1 is running LACP, that both interfaces are successfully bundled in the port channel, that the port channel is functioning at Layer 2, and that it is in use. On the other hand, Port-Channel 2 is running PAgP, both interfaces are also successfully bundled in the port channel, and the port channel is being used as a Layer 3 link between DSW1 and CSW1.

The **show interfaces Port-channel 1** command output displays the cumulative bandwidth (2 Gbps) of the virtual link and confirms which physical interfaces are part of the EtherChannel bundle.

The **show lacp neighbor** and **show pagp neighbor** commands produce similar output regarding DSW1's EtherChannel neighbors: ports used, device ID, control packet interval, and flags indicating whether slow or fast hellos are in use.

Advanced EtherChannel Tuning

It is possible to tune LACP to further improve the overall behavior of an EtherChannel. This section looks at some of the commands available to override LACP default behavior.

LACP Hot-Standby Ports

When LACP is enabled, the software, by default, tries to configure the maximum number of LACP-compatible ports in a channel, up to a maximum of 16 ports. Only 8 LACP links can be active at one time; the remaining 8 links are placed in hot-standby mode. If one of the active links becomes inactive, a link that is in hot-standby mode becomes active in its place. This is achieved by specifying the maximum number of active ports in a channel, in which case the remaining ports become hot-standby ports. For example, if you specify a maximum of 5 ports in a channel, up to 11 ports become hot-standby ports.

If you configure more than 8 links for an EtherChannel group, the software automatically decides which of the hot-standby ports to make active, based on the LACP priority. To every link between systems that operate LACP, the software assigns a unique priority made up of these elements (in priority order):

- LACP system priority
- System ID (the device MAC address)

- LACP port priority
- Port number

In priority comparisons, numerically lower values have higher priority. The priority determines which ports should be put in standby mode when there is a hardware limitation that prevents all compatible ports from aggregating.

Determining which ports are active and which are hot standby is a two-step procedure. First, the system with a numerically lower system priority and system ID is placed in charge of the decision. Next, that system decides which ports are active and which are hot standby, based on its values for port priority and port number. The port priority and port number values for the other system are not used.

You can change the default values of the LACP system priority and the LACP port priority to affect how the software selects active and standby links.

Configuring the LACP Max Bundle Feature

When you specify the maximum number of bundled LACP ports allowed in a port channel, the remaining ports in the port channel are designated as hot-standby ports. Use the **lacp max-bundle** port channel interface command, as shown in Example 27-5. Since DSW1 currently has two interfaces in Port-channel 1, by setting a maximum of 1, one port is placed in hot-standby mode.

Example 27-5 Configuring the LACP Max Bundle Feature

```
DSW1(config)# interface Port-channel 1
DSW1(config-if)# lacp max-bundle 1
```

```
DSW1# show etherchannel summary
Flags:   D - down         P - bundled in port-channel
         I - stand-alone  cs - suspended
         H - Hot-standby (LACP only)
         R - Layer3       S - Layer2
         U - in use       N - not in use, no aggregation
         f - failed to allocate aggregator

         M - not in use, minimum links not met
         m - not in use, port not aggregated due to minimum links not met
         u - unsuitable for bundling
         w - waiting to be aggregated
         d - default port

         A - formed by Auto LAG
```

```
Number of channel-groups in use: 2
Number of aggregators:           2

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(SU)         LACP      Gi1/0/1(P)  Gi1/0/2(H)
2      Po2(RU)         PAgP      Gi1/0/3(P)  Gi1/0/4(P)
```

Notice that DSW1 has placed Gi1/0/2 in hot-standby mode. Both Gi1/0/1 and Gi1/0/2 ports have the same default LACP port priority of 32768, so the LACP master switch chooses the higher-numbered port to be the candidate for hot-standby mode.

Configuring the LACP Port Channel Min-Links Feature

You can specify the minimum number of active ports that must be in the link-up state and bundled in an EtherChannel for the port channel interface to transition to the link-up state. Using the **port-channel min-links** port channel interface command, you can prevent low-bandwidth LACP EtherChannels from becoming active. Port channel min-links also cause LACP EtherChannels to become inactive if they have too few active member ports to supply the required minimum bandwidth.

Configuring the LACP System Priority

You can configure the system priority for all the EtherChannels that are enabled for LACP by using the **lacp system-priority** command in global configuration mode. You cannot configure a system priority for each LACP-configured channel. By changing this value from the default, you can affect how the software selects active and standby links. A lower value is preferred to select which switch is the master for the port channel. Use the **show lacp sys-id** command to view the current system priority.

Configuring the LACP Port Priority

By default, all ports use the same default port priority of 32768. If the local system has a lower value for the system priority and the system ID than the remote system, you can affect which of the hot-standby links become active first by changing the port priority of LACP EtherChannel ports to a lower value than the default. The hot-standby ports that have lower port numbers become active in the channel first. You can use the **show etherchannel summary** privileged EXEC command to see which ports are in the hot-standby mode (denoted with an H port-state flag). Use the **lacp port-priority** command in interface configuration mode to set a value between 1 and 65535. For instance, in Example 27-5, if the LACP port priority were lowered for interface Gi1/0/2, the other interface in the bundle (Gi1/0/1) would take over the hot-standby role instead.

Configuring LACP Fast Rate Timer

You can change the LACP timer rate to modify the duration of the LACP timeout. Use the **lacp rate {normal | fast}** command to set the rate at which LACP control packets are received by an LACP-supported interface. You can change the timeout rate from the default rate (30 seconds) to the fast rate (1 second). This command is supported only on LACP-enabled interfaces.

Example 27-6 illustrates the configuration and verification of the LACP system priority, LACP port priority, and LACP fast rate timer.

Example 27-6 Configuring and Verifying LACP System Priority, LACP Port Priority, and LACP Fast Rate Timer

```
DSW1(config)# lacp system-priority 20000
DSW1(config)# interface GigabitEthernet 1/0/2
DSW1(config-if)# lacp port-priority 100
DSW1(config-if)# interface range GigabitEthernet 1/0/1-2
DSW1(config-if-range)# lacp rate fast
```

```
DSW1# show lacp internal
Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode

Channel group 1
                            LACP port     Admin     Oper    Port        Port
Port      Flags   State     Priority      Key       Key     Number      State
Gi1/0/1   FA      hot-sby   32768         0x1       0x1     0x102       0x3F
Gi1/0/2   FA      bndl      100           0x1       0x1     0x103       0xF
```

```
DSW1# show lacp sys-id
20000, aabb.cc80.0100
```

In the output, the **F** flag indicates that both Gi1/0/1 and Gi1/0/2 are using fast LACP packets. Since the port priority was lowered to 100 on Gi1/0/2, Gi1/0/1 is now in hot-standby mode. Also, the system priority was lowered on DSW1 to a value of 20000.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	5
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	1

Day 26

EIGRP

ENCOR 350-401 Exam Topics

Infrastructure

- Layer 3
  - Compare routing concepts of EIGRP and OSPF (advanced distance vector vs. linked state, load balancing, path selection, path operations, metrics)

Key Topics

Today we review key concepts related to Enhanced Interior Gateway Routing Protocol (EIGRP). EIGRP is advanced compared to traditional distance vector–style dynamic routing protocols (such as RIP and IGRP). The primary purposes of EIGRP are to maintain stable routing tables on Layer 3 devices and quickly discover alternate paths in the event of a topology change. Cisco designed EIGRP as a migration path from the proprietary IGRP protocol to solve some of IGRP's deficiencies and as a solution to support multiple routed protocols. The protocols it supports today include IPv4, IPv6, VoIP dial plans, and Cisco Performance Routing (PfR) via Service Advertisement Framework (SAF). It previously supported the now-defunct IPX and AppleTalk routed protocols. Even though these protocols are no longer used, EIRGP's solution for networks in the late 1990s and early 2000s provided benefits compared to using OSPFv2, as OSPFv2 supports only IPv4. While EIGRP was initially proprietary, parts of the EIGRP protocol are now an open standard, defined in RFC 7868.

EIGRP Features

EIGRP combines the advantages of link-state routing protocols such as OSPF and IS-IS and distance vector routing protocols such as RIP. EIGRP may act like a link-state routing protocol because it uses a Hello protocol to discover neighbors and form neighbor relationships, and only partial updates are sent when a change occurs. However, EIGRP is based on the key distance vector routing protocol principle, in which information about the rest of the network is learned from directly connected neighbors.

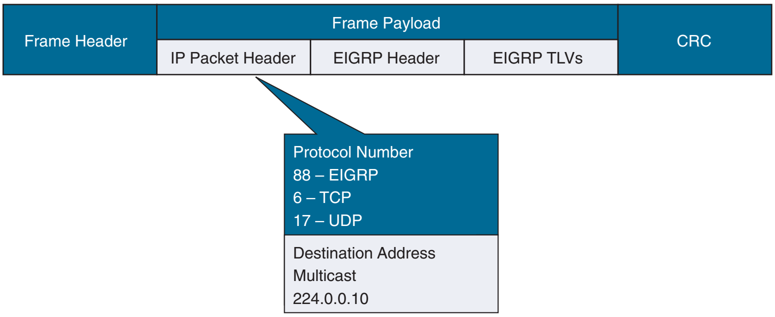
The following are some of the important features of EIGRP:

- **Rapid convergence:** EIGRP uses the diffusing update algorithm (DUAL) to achieve rapid convergence. As the computational engine that runs EIGRP, DUAL resides at the center of the routing protocol, guaranteeing loop-free paths and backup paths throughout the routing domain. A router that uses EIGRP stores all available backup routes for destinations so that it can quickly adapt to alternate routes. If the primary route in the routing table fails, the best backup route is immediately added to the routing table. If no appropriate route or backup route exists in the local routing table, EIGRP queries its neighbors to discover an alternate route.
- **Load balancing:** EIGRP supports equal-metric load balancing (also called *equal-cost multipathing* [ECMP]) and unequal-metric load balancing to allow administrators to better distribute traffic flow in their networks.
- **Loop-free, classless routing protocol:** Because EIGRP is a classless routing protocol, it advertises a routing mask for each destination network. The routing mask feature enables EIGRP to support discontiguous subnetworks and VLSM.
- **Multi-address family support:** EIGRP supports multiple routed protocols. It has always supported IPv4, and in the past it also supported protocols such as IPX and AppleTalk (which are now deprecated). Today multi-address family support means EIGRP is ready for IPv6. EIGRP can also be used for a solution to distribute dial plan information within a large-scale VoIP network by integrating with Cisco Unified Communications Manager and for Cisco PfR.
- **Reduced bandwidth use:** EIGRP updates can be thought of as either partial or bounded. EIGRP does not make periodic updates. The term *partial* refers to an update that includes only information about the route changes. EIGRP sends these incremental updates when the state of a destination changes instead of sending the entire contents of the routing table. The term *bounded* refers to the propagation of partial updates that are sent only to routers that the changes affect. By sending only the routing information that is needed and only to routers that need it, EIGRP minimizes the bandwidth required to send EIGRP updates. EIGRP uses multicast and unicast rather than broadcast. Multicast EIGRP packets use the reserved multicast address 224.0.0.10. As a result, end stations are unaffected by routing updates and requests for topology information.

EIGRP Reliable Transport Protocol

As illustrated in Figure 26-1, EIGRP runs directly above the IP layer as its own protocol, numbered 88. RTP is the component of EIGRP that is responsible for guaranteed, ordered delivery of EIGRP packets to all neighbors. It supports intermixed transmission of multicast or unicast packets. When using multicast on the segment, packets are sent to the reserved multicast address 224.0.0.10 for IPv4 and FF00::A for IPv6.

Figure 26-1 EIGRP Encapsulation



EIGRP Operation Overview

Operation of the EIGRP protocol is based on information stored in three tables: the neighbor table, the topology table, and the routing table. The main information that is stored in the neighbor table is a set of neighbors with which the EIGRP router has established adjacencies. A neighbor is characterized by its primary IP address and the directly connected interface that leads to it.

The topology table contains all destination routes advertised by the neighbor routers. Each entry in the topology table is associated with a list of neighbors that have advertised the destination. For each neighbor, an advertised metric is recorded. This value is the metric that a neighbor stores in its routing table to reach a particular destination. Another important piece of information is the metric that the router itself uses to reach the same destination. This value is the sum of the advertised metric from the neighbor plus the link cost to the neighbor. The route with the best metric to the destination is called the *successor*, and it is placed in the routing table and advertised to the other neighbors. EIGRP uses the terms *successor route* and *feasible successor* when referring to the best path and the backup path:

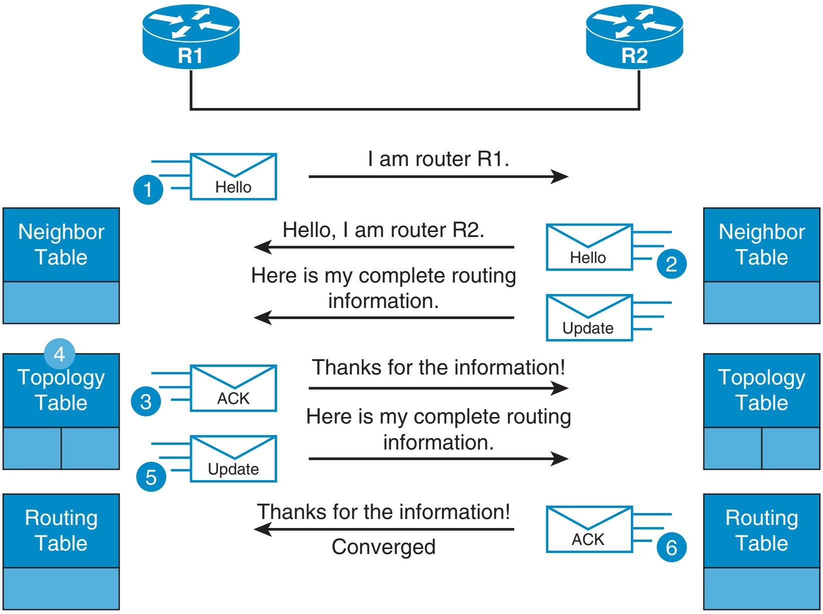
- The EIGRP successor route is the lowest-metric best path to reach a destination. EIGRP successor routes are placed into the routing table.
- The feasible successor (FS) is the best alternative loop-free backup path to reach a destination. Because it is not the least-cost or lowest-metric path, it is not selected as the primary path to forward packets, and it is not inserted into the routing table. Feasible successors are important as they allow an EIGRP router to recover immediately after a network failure.

With EIGRP, the processes to establish and discover neighbor routes occur simultaneously. A high-level description of the process follows, using the topology in Figure 26-2:

1. R1 comes up on the link and sends a hello packet through all its EIGRP-configured interfaces.
2. R2 receives the hello packet on one interface and replies with its own hello packet and an update packet that contains the routes in the routing tables that were not learned through that interface (split horizon). R2 sends an update packet to R1, but a neighbor relationship is not established until R2 sends a hello packet to R1. The update packet from R2 has the initialization bit set, indicating that this interaction is the initialization process. The update packet includes information about the routes that the neighbor (R2) is aware of, including the metric that the neighbor is advertising for each destination.

3. After both routers have exchanged hellos and the neighbor adjacency is established, R1 replies to R2 with an ACK packet, indicating that it received the update information.
4. R1 assimilates all the update packets into its topology table. The topology table includes all destinations that are advertised by neighboring adjacent routers. It lists each destination, all the neighbors that can reach the destination, and their associated metrics.
5. R1 sends an update packet to R2.
6. Upon receiving the update packet, R2 sends an ACK packet to R1.

Figure 26-2 EIGRP Operation Overview


R1
R2
1
Hello
I am router R1.
Neighbor Table
Hello, I am router R2.
Hello
2
Neighbor Table
Here is my complete routing information.
Update
4
Topology Table
3
ACK
Thanks for the information!
Topology Table
Here is my complete routing information.
5
Update
Routing Table
Thanks for the information!
Converged
ACK
6
Routing Table

EIGRP Packet Format

EIGRP sends a number of packet types, as shown in Table 26-1.

Table 26-1 EIGRP Packets

Packet Type	Description
Hello	Used to discover a neighbor before establishing an adjacency. EIGRP hello packets are sent as multicasts and contain acknowledgment number 0. EIGRP routers must form neighbor relationships before exchanging EIGRP updates.
Update	Used to communicate the routes that a particular router has used to converge. EIGRP updates are sent as multicasts when a new route is discovered or when convergence is completed; they are sent as unicasts when synchronizing topology tables with new neighbors at EIGRP startup. These packets are sent reliably between EIGRP routers.

Packet Type	Description
Query	Used to query other EIGRP neighbors for a feasible successor when EIGRP is recomputing a route in which the router does not have a feasible successor. EIGRP queries are sent reliably as multicasts.
Requests	Request packets are used to get specific information from one or more neighbors. Request packets are used in route server applications. EIGRP requests can be multicast or unicast. Requests are transmitted unreliably.
Reply	Sent as the response to an EIGRP query packet. EIGRP replies are sent reliably as unicasts.
Acknowledge	Used to acknowledge EIGRP updates, queries, and replies; hello and ACK packets do not require acknowledgment. ACKs are hello packets that contain no data and a nonzero acknowledgment number, and they are sent as unicasts.

An EIGRP query packet is sent by a router to advertise that a route is in active state and the originator is requesting alternate path information from its neighbors. A route is considered passive when the router is not performing recomputation for that route, and a route is considered active when the router is performing recomputation to seek a new successor when the existing successor has become invalid.

Establishing EIGRP Neighbor Adjacency

Establishing a neighbor relationship or adjacency with EIGRP is less complicated than with Open Shortest Path First (OSPF), but the process does need to follow certain rules. The following parameters should match in order for EIGRP to create a neighbor adjacency:

- **AS number:** An EIGRP router establishes neighbor relationships (adjacencies) only with other routers within the same autonomous system. An EIGRP autonomous system number is a unique number established by an enterprise. It is used to identify a group of devices and enables that system to exchange interior routing information with other neighboring routers in the same autonomous system.
- **K values (metric):** EIGRP K values are the metrics that EIGRP uses to calculate routes. Mismatched K values can prevent neighbor relationships from being established and can negatively impact network convergence. A message like the following is logged at the console when this occurs:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.4.1.5 (GigabitEthernet0/1) is
down: K-value mismatch
```

- **Common subnet:** EIGRP cannot form neighbor relationships using secondary addresses, as only primary addresses are used as the source IP addresses of all EIGRP packets. A message like the following is logged at the console when neighbors are configured on different subnets:

```
IP-EIGRP(Default-IP-Routing-Table:1): Neighbor 10.1.1.2 not on common subnet
for GigabitEthernet0/1
```

- **Authentication method and password:** Regarding authentication, EIGRP becomes a neighbor with any router that sends a valid hello packet. Due to security considerations, this open aspect requires filtering to limit peering to valid routers only. Filtering ensures that only authorized routers exchange routing information within an autonomous system. A message like the following is logged at the console if authentication is incorrectly configured:

```
EIGRP: GigabitEthernet0/1: ignored packet from 10.1.1.3, opcode = 1 (missing
authentication or key-chain missing)
```

All this information is contained in the EIGRP hello message. If a router running EIGRP receives a hello message from a new router and the preceding parameters match, a new adjacency is formed. Note that certain parameters that are key in the neighbor adjacency process of OSPF are not present in this list. For instance, EIGRP doesn't care if the hello timers between neighbors are mismatched. OSPF doesn't have a designation for an autonomous system number even though the concept of an AS is important in the implementation of OSPF. The process ID used in OSPF is a value that is only locally significant to a particular router.

The **passive-interface** command in EIGRP suppresses the exchange of hello packets between two routers, which results in the loss of their neighbor relationship and the suppression of incoming routing packets.

EIGRP Metrics

Unlike other routing protocols (such as RIP and OSPF), EIGRP does not use a single attribute in the metric for its routes. EIGRP uses a combination of four different elements related to physical characteristics of an interface to determine the metric for its routes:

- **Bandwidth (K1):** This value is the smallest bandwidth of all outgoing interfaces between the source and destination, in kilobits per second.
- **Load (K2):** This value represents the worst load on a link between the source and destination, which is computed based on the packet rate and the configured bandwidth of the interface.
- **Delay (K3):** The is the sum of all interface delay along the path, in tens of microseconds.
- **Reliability (K4, K5):** These values represent the worst reliability between the source and destination, based on keepalives.

EIGRP monitors metric weights by using K values on an interface to allow the tuning of EIGRP metric calculations. K values are integers from 0 to 128; these integers, in conjunction with variables such as bandwidth and delay, are used to calculate the overall EIGRP composite cost metric. EIGRP default K values have been carefully selected to provide optimal performance in most networks.

The EIGRP composite metric is calculated using the formula shown in Figure 26-3.

By default, K1 (bandwidth) and K3 (delay) are set to 1. K2, K4, and K5 are set to 0. The result is that only the bandwidth and delay values are used in the computation of the default composite metric, as shown in Figure 26-4.

Figure 26-3 EIGRP Metric Formula

Total sum of all delays in microseconds (μs)

$$Metric = \left[\left(K1 * BWmin + \frac{K2 * BWmin}{256 - Load} + K3 * \frac{\sum Delay}{10}\right) * \frac{K5}{K4 + Reliability}\right] * 256$$

$$BWmin = \frac{10^7}{least\_bandwidth}$$

Figure 26-4 EIGRP Simplified Metric Calculation

$$Metric = \left[\left(\frac{10^7}{least\_bandwidth} + \frac{cumulative\_delay}{10}\right)\right] * 256$$

The 256 multiplier in the formula is based on one of the original goals of EIGRP: to offer enhanced routing solutions over legacy IGRP. To achieve this, EIGRP uses the same composite metric as IGRP, and the terms are multiplied by 256 to change the metric from 24 bits to 32 bits.

By using the **show interfaces** command, you can examine the actual values that are used for bandwidth, delay, reliability, and load in the computation of the routing metric. The output in Example 26-1 shows the values that are used in the composite metric for the Serial0/0/0 interface.

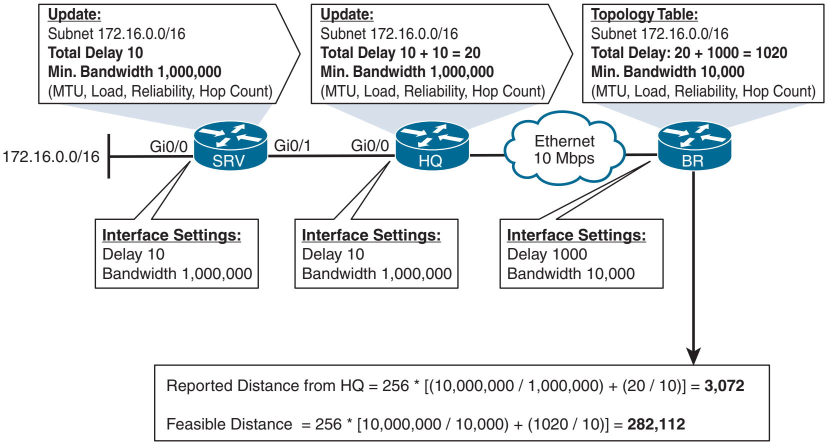
Example 26-1 Verifying Interface Metrics

```
R1# show interfaces Serial0/0/0
Serial0/0/0 is up, line protocol is up
  Hardware is GT96K Serial
  Description: Link to HQ
  MTU 1500 bytes, BW 1544 Kbit/sec, DLY 20000 usec,
    reliability 255/255, txload 1/255, rxload 1/255
<... output omitted ...>
```

You can influence the EIGRP metric by changing bandwidth and delay on an interface, using the **bandwidth** *kbps* and **delay** *tens-of-microseconds* interface configuration commands. However, when performing path manipulation in EIGRP, changing the delay is preferred. Because EIGRP uses the lowest bandwidth in the path, changing the bandwidth may not change the metric. Changing the bandwidth value might create other problems, such as altering the operation of features such as QoS and affecting the telemetry data in monitoring.

Figure 26-5 illustrates a simple topology using EIGRP. The 172.16.0.0/16 subnet is advertised by SRV to HQ using a delay of 10 µs and a minimum bandwidth of 1,000,000 Kbps because the local interface used to reach that subnet is a Gigabit Ethernet interface. HQ then advertises the 172.16.0.0/16 prefix with a cumulative delay of 20 µs (10 µs for the SRV Gi0/0 interface and 10 µs for the HQ Gi0/0 interface) and a minimum bandwidth of 1,000,000 Kbps. The BR router calculates a reported distance (RD) of 3072, based on the information learned from the HQ router. The BR router then calculates its own feasible distance (FD) based on a cumulative delay of 1020 µs (10 µs + 10 µs + 1000 µs for the local interface on BR). Also, the minimum bandwidth is now 10,000 Kbps because the BR router is connected to an Ethernet WAN cloud. The calculated FD is 282,112 for BR to reach the 172.16.0.0/16 subnet hosted on the SRV router. Note that, although not shown in Figure 26-5, both SRV and HQ would also calculate RDs and FDs to reach the 172.16.0.0/16 subnet. RD and FD are explained in more detail later today.

Figure 26-5 EIGRP Attribute Propagation


Update:
Subnet 172.16.0.0/16
Total Delay 10
Min. Bandwidth 1,000,000
(MTU, Load, Reliability, Hop Count)
Update:
Subnet 172.16.0.0/16
Total Delay 10 + 10 = 20
Min. Bandwidth 1,000,000
(MTU, Load, Reliability, Hop Count)
Topology Table:
Subnet 172.16.0.0/16
Total Delay: 20 + 1000 = 1020
Min. Bandwidth 10,000
(MTU, Load, Reliability, Hop Count)
172.16.0.0/16
Gi0/0
SRV
Gi0/1
Gi0/0
HQ
Ethernet
10 Mbps
BR
Interface Settings:
Delay 10
Bandwidth 1,000,000
Interface Settings:
Delay 10
Bandwidth 1,000,000
Interface Settings:
Delay 1000
Bandwidth 10,000
Reported Distance from HQ = 256 * [(10,000,000 / 1,000,000) + (20 / 10)] = 3,072
Feasible Distance = 256 * [10,000,000 / 10,000) + (1020 / 10)] = 282,112

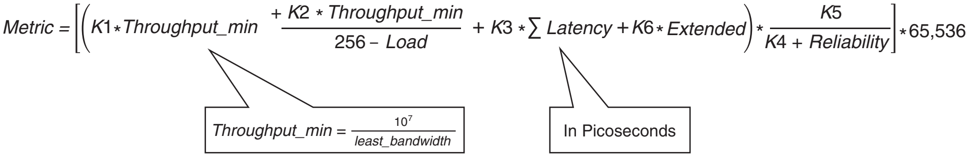
EIGRP Wide Metrics

The EIGRP composite cost metric (calculated using the bandwidth, delay, reliability, load, and K values) is not scaled correctly for high-bandwidth interfaces or EtherChannels, and this results in incorrect or inconsistent routing behavior. The lowest delay that can be configured for an interface is 10 microseconds. As a result, high-speed interfaces, such as 10 Gigabit Ethernet (GE) interfaces, or high-speed interfaces channeled together (Gigabit Ethernet EtherChannel) appear to EIGRP as a single GigabitEthernet interface. This may cause undesirable equal-cost load balancing. To resolve this issue, the EIGRP Wide Metrics feature supports 64-bit metric calculations and Routing Information Base (RIB) scaling that provides the ability to support interfaces (either directly or via channeling techniques such as EtherChannel) up to approximately 4.2 Tbps.

To accommodate interfaces with bandwidths above 1 Gbps and up to 4.2 Tbps and to allow EIGRP to perform correct path selections, the EIGRP composite cost metric formula is modified. The paths are selected based on the computed time. The time that information takes to travel through links is measured in picoseconds. The interfaces can be directly capable of these high speeds, or the interfaces can be bundles of links with an aggregate bandwidth greater than 1 Gbps.

Figure 26-6 illustrates the EIGRP Wide Metrics formula, which is scaled by 65,536 instead of 256.

Figure 26-6 EIGRP Wide Metrics Formula



$$Metric = \left[\left(K1 * Throughput\_min \frac{+ K2 * Throughput\_min}{256 - Load} + K3 * \sum Latency + K6 * Extended\right) * \frac{K5}{K4 + Reliability}\right] * 65{,}536$$

The default K values are as follows:

K1 = K3 = 1

K2 = K4 = K5 = 0

K6 = 0

The EIGRP Wide Metrics feature also introduces K6. K6 allows for extended attributes, which can be used for higher aggregate metrics than those having lower energy usage. Currently there are two extended attributes, **jitter** and **energy**, which can be used to reflect in paths with a higher aggregate metric than those having lower energy usage.

By default, the path selection scheme used by EIGRP is a combination of throughput (rate of data transfer) and latency (time taken for data transfer, in picoseconds).

For IOS interfaces that do not exceed 1 Gbps, the delay value is derived from the reported interface delay, converted to picoseconds:

$$Delay = \left(Interface\ Delay * 10^6\right)$$

Beyond 1 Gbps, IOS does not report delays properly, so a computed delay value is used:

$$Delay = \left(\frac{10^7 * 10^6}{Interface\ \ Bandwidth}\right)$$

Latency is calculated based on the picosecond delay values and scaled by 65,536:

$$Latency = \frac{Delay * 65{,}536}{10^6}$$

Similarly, throughput is calculated based on the worst bandwidth in the path, in Kbps, and scaled by 65,536:

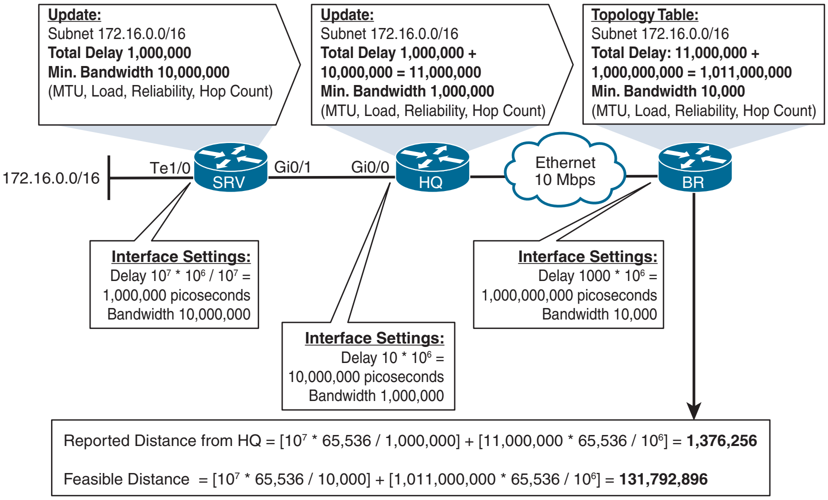
$$Throughput\_min = \frac{10^7 * 65{,}536}{Bandwidth}$$

The simplified formula for calculating the composite cost metric is as follows:

$$Composite\ \ Cost\ Metric = Throughput\_min + \sum Latency$$

Figure 26-7 uses the same topology as Figure 26-5, but the interface on the SRV router connected to the 172.16.0.0/16 subnet has been changed to a 10 Gigabit Ethernet interface, and the Wide Metrics feature is used in the metric calculation. Notice that the picosecond calculation is different for the 10 Gigabit Ethernet interface than for the Gigabit Ethernet interface discussed earlier.

Figure 26-7 EIGRP Wide Metrics Attribute Propagation


Update:
Subnet 172.16.0.0/16
Total Delay 1,000,000
Min. Bandwidth 10,000,000
(MTU, Load, Reliability, Hop Count)
Update:
Subnet 172.16.0.0/16
Total Delay 1,000,000 + 10,000,000 = 11,000,000
Min. Bandwidth 1,000,000
(MTU, Load, Reliability, Hop Count)
Topology Table:
Subnet 172.16.0.0/16
Total Delay: 11,000,000 + 1,000,000,000 = 1,011,000,000
Min. Bandwidth 10,000
(MTU, Load, Reliability, Hop Count)
172.16.0.0/16
Te1/0
SRV
Gi0/1
Gi0/0
HQ
Ethernet 10 Mbps
BR
Interface Settings:
Delay $10^7 * 10^6 / 10^7$ = 1,000,000 picoseconds
Bandwidth 10,000,000
Interface Settings:
Delay $10 * 10^6$ = 10,000,000 picoseconds
Bandwidth 1,000,000
Interface Settings:
Delay $1000 * 10^6$ = 1,000,000,000 picoseconds
Bandwidth 10,000
Reported Distance from HQ = [10^7 * 65,536 / 1,000,000] + [11,000,000 * 65,536 / 10^6] = 1,376,256
Feasible Distance = [10^7 * 65,536 / 10,000] + [1,011,000,000 * 65,536 / 10^6] = 131,792,896

With the calculation of larger bandwidths, EIGRP can no longer fit the computed metric into a 4-byte unsigned long value that is needed by the Cisco RIB. To set the RIB scaling factor for EIGRP, use the **metric rib-scale** command. When you configure the **metric rib-scale** command, all EIGRP routes in the RIB are cleared and replaced with the new metric values. The default value is 128. Example 26-2 shows how to use the **show ip protocols**, **show ip route eigrp**, and **show ip eigrp topology** commands to verify how the router is using the EIGRP Wide Metrics feature to calculate the composite metric for a route.

Note that the 64-bit metric calculations work only in EIGRP named mode configurations. EIGRP classic mode uses 32-bit metric calculations.

Example 26-2 Verifying EIGRP Wide Metrics Calculations

```
BR# show ip protocols
<. . . output omitted . . .>
Routing Protocol is "eigrp 10"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Default networks flagged in outgoing updates
  Default networks accepted from incoming updates
  EIGRP-IPv4 VR(TEST) Address-Family Protocol for AS(10)
    Metric weight K1=1, K2=0, K3=1, K4=0, K5=0 K6=0
    Metric rib-scale 128
    Metric version 64bit
    Soft SIA disabled
    NSF-aware route hold timer is 240
    Router-ID: 10.2.2.2
    Topology : 0 (base)
      Active Timer: 3 min
      Distance: internal 90 external 170
      Maximum path: 4
      Maximum hopcount 100
      Maximum metric variance 1
      Total Prefix Count: 3
      Total Redist Count: 0
<. . . output omitted . . .>
```

```
BR# show ip route eigrp
<. . . output omitted . . .>
D         172.16.0.0/16 [90/1029632] via 10.2.2.1, 00:53:35, Ethernet0/1
```

```
BR# show ip eigrp topology 172.16.0.0/16
EIGRP-IPv4 VR(TEST) Topology Entry for AS(10)/ID(10.2.2.2) for 172.16.0.0/16
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 131792896, RIB
  is 1029632
  Descriptor Blocks:
  10.2.2.1 (Ethernet0/1), from 10.2.2.1, Send flag is 0x0
      Composite metric is (131792896/1376256), route is Internal
      Vector metric:
        Minimum bandwidth is 10000 Kbit
        Total delay is 1011000000 picoseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
        Originating router is 10.1.1.1
```

In Example 26-2, the output from the **show ip protocols** command confirms the rib-scale value and the 64-bit metric version, as well as the default K values (including K6). The **show ip route eigrp** command displays the scaled-down version of the calculated metric (131792896 / 128 = 1029632) for the 172.16.0.0/16 prefix. The **show ip eigrp topology** command confirms the minimum bandwidth (10,000 Kbps) and total delay (1011000000 picoseconds) used to calculate the metric, as well as the FD (131792896) and RD (1376256) for the route.

EIGRP Path Selection

In the context of dynamic IP routing protocols such as EIGRP, the term *path selection* refers to the method by which the protocol determines the best path to a destination IP network.

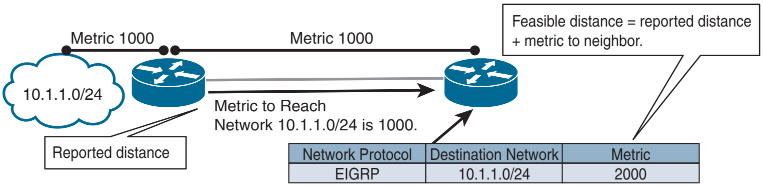
Each EIGRP router maintains a neighbor table that includes a list of directly connected EIGRP routers that have formed an adjacency with this router. Upon creating an adjacency, an EIGRP router exchanges topology data and runs the path selection process to determine the current best path(s) to each network. After the exchange of topology, the hello process continues to run to track neighbor relationships and to verify the status of these neighbors. As long as a router continues to hear EIGRP neighbor hellos, it knows that the topology is currently stable.

In a dual-stack environment with networks running both IPv4 and IPv6, each EIGRP router maintains a separate neighbor table and topology table for each routed protocol. The topology table includes route entries for every destination that the router learns from its directly connected EIGRP neighbors. EIGRP chooses the best routes to a destination from the topology table and submits them to the routing engine for consideration. If the EIGRP route is the best option, it is installed into the routing table. It is possible that the router has a better path to the destination already, as determined by administrative distance, such as a static route.

EIGRP uses two parameters to determine the best route (successor) and any backup routes (feasible successors) to a destination, as shown in Figure 26-8:

- **Reported distance (RD):** The EIGRP metric for an EIGRP neighbor to reach a destination network.
- **Feasible distance (FD):** The EIGRP metric for a local router to reach a destination network. In other words, it is the sum of the reported distance of an EIGRP neighbor and the metric to reach that neighbor. This sum provides an end-to-end metric from the router to the remote network.

Figure 26-8 EIGRP Feasible Distance and Reported Distance



Loop-Free Path Selection

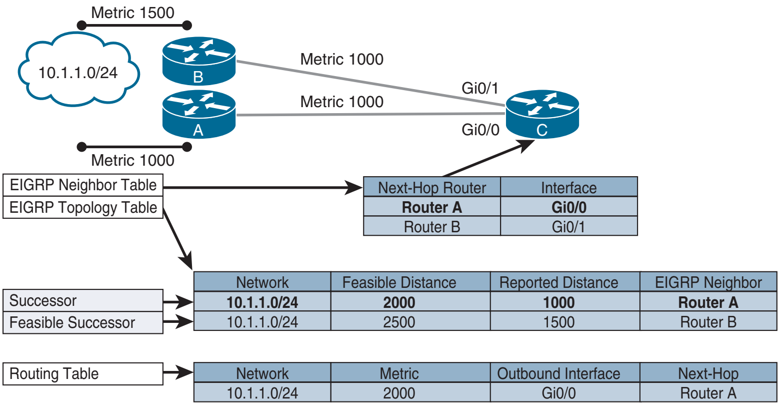
EIGRP uses the DUAL finite-state machine to track all routes advertised by all neighbors with the topology table, performs route computation on all routes to select an efficient and loop-free path to all destinations, and inserts the lowest-metric route into the routing table.

A router compares all FDs to reach a specific network and then selects the lowest FD as the best path, and it then submits this path to the routing engine for consideration. Unless this route has already been submitted with a lower administrative distance, this path is installed into the routing table. The FD for the chosen route becomes the EIGRP routing metric to reach this network in the routing table.

The EIGRP topology database contains all the routes that are known to each EIGRP neighbor. As shown in Figure 26-9, Routers A and B sent their routing information to Router C, whose table is displayed. Both Routers A and B have routes to network 10.1.1.0/24 and to other networks that are not shown.

Router C has two entries to reach 10.1.1.0/24 in its topology table. The EIGRP metric for Router C to reach both Routers A and B is 1000. Add this metric (1000) to the respective RD for each router, and the results represent the FDs that Router C must use to reach network 10.1.1.0/24.

Figure 26-9 EIGRP Path Selection



Next-Hop Router	Interface
Router A	**Gi0/0**
Router B	Gi0/1

Network	Feasible Distance	Reported Distance	EIGRP Neighbor
10.1.1.0/24	**2000**	**1000**	**Router A**
10.1.1.0/24	2500	1500	Router B

Network	Metric	Outbound Interface	Next-Hop
10.1.1.0/24	2000	Gi0/0	Router A

Router C chooses the smallest FD (2000) and installs it in the IP routing table as the best route to reach 10.1.1.0/24. The route with the smallest FD that is installed in the routing table is called the *successor route*.

Router C then chooses a backup route to the successor—called a *feasible successor route*—if one or more feasible successor routes exist. To become a feasible successor, a route must satisfy the feasibility condition: A next-hop router must have an RD that is less than the FD of the current successor route (and, therefore, the route is tagged as a feasible successor). This rule is used to ensure that the network is loop free. In Figure 26-9, the RD from Router B is 1500, and the current FD is 2000, so the path through Router B meets the feasibility condition and is installed as feasible successor.

If the route via the successor becomes invalid, possibly because of a topology change or because a neighbor changes the metric, DUAL checks for feasible successors to the destination route. If a feasible successor is found, DUAL uses it, avoiding the need to recompute the route. A route changes from a passive state to an active state if no feasible successor exists, and a DUAL computation must occur to determine the new successor.

Keep in mind that each routing protocol uses the concept of administrative distance (AD) when choosing the best path between multiple routing sources. A route with a lower value is always preferred. EIGRP has AD of 90 for internal routes, AD of 170 for external routes, and AD of 5 for summary routes.

EIGRP Load Balancing and Sharing

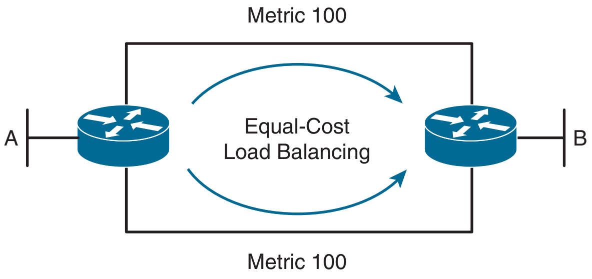
In general, load balancing is the capability of a router to distribute traffic over all the router network ports that are within the same distance of the destination address. Load balancing increases the utilization of network segments and, in this way, increases effective network bandwidth. Equal-cost multi-path (ECMP) is supported by routing in general via the **maximum-paths** command. This command can be used with EIGRP, OSPF, and RIP. The default value and possible range vary between IOS versions and devices. Use the **show ip protocols** command to verify the currently configured value. EIGRP is unique among routing protocols, as it supports both equal- and unequal-cost load balancing. Route-based load balancing is done on a per-flow basis, not per packet.

ECMP is a routing strategy in which next-hop packet forwarding to a single destination can occur over multiple "best paths" that tie for top place in routing metric calculations.

Equal-Cost Load Balancing

Given that good network design involves Layer 3 path redundancy, it is a common customer expectation that if there are multiple devices and paths to a destination, all paths should be utilized. In Figure 26-10, Networks A and B are connected with two equal-cost paths. For this example, assume that the links are Gigabit Ethernet.

Figure 26-10 EIGRP Equal-Cost Load Balancing

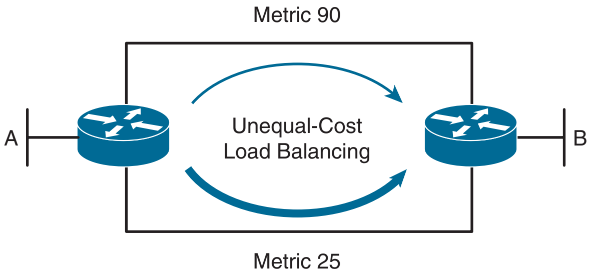


Equal-cost load balancing is the ability of a router to distribute traffic over all its network ports that are the same metric from the destination address. Load balancing increases the use of network segments and increases effective network bandwidth. By default, Cisco IOS Software applies load balancing across up to four equal-cost paths for a certain destination IP network, if such paths exist. With the **maximum-paths** router configuration command, you can specify the number of routes that can be kept in the routing table. If you set the value to 1, you disable load balancing.

Unequal-Cost Load Balancing

EIGRP can balance traffic across multiple routes that have different metrics. This type of balancing is called *unequal-cost load balancing*. In Figure 26-11, there is a cost difference of almost 4:1 between the paths. A real-world example of such a situation is the case of a WAN connection from HQ to a branch, with a 6-Mbps MPLS link as the primary WAN link and a T1 (1.544 Mbps) backup link.

Figure 26-11 EIGRP Unequal-Cost Load Balancing



You can use the **variance** command to tell EIGRP to install routes in the routing table, as long as they are less than the current best cost multiplied by the variance value. In the example in Figure 26-11, setting the variance to 4 would allow EIGRP to install the backup path and send traffic over it. The backup path is now performing work instead of just idling. The default variance is equal to 1, which disables unequal-cost load balancing.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	7
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	4

This page intentionally left blank

Day 25

OSPFv2

ENCOR 350-401 Exam Topics

Infrastructure

- Layer 3
  - Configure and verify simple OSPF environments, including multiple normal areas, summarization, and filtering (neighbor adjacency, point-to-point and broadcast network types, and passive interface)

Key Topics

Today we start our review of the Open Shortest Path First (OSPF) routing protocol. OSPF is a vendor-agnostic link-state routing protocol that builds and maintains the routing tables needed for IPv4 and IPv6 traffic. Today we focus on OSPFv2 (RFC 2328), which works only with IPv4. The most recent implementation of OSPF, OSPFv3, works with both IPv4 and IPv6. OSPFv3 is discussed on Day 24, "Advanced OSPFv2 and OSPFv3." Both versions of OSPF are open standards and can run on various devices that need to manage routing tables. Devices such as traditional routers, multilayer switches, servers, and firewalls can benefit from running OSPF. The shortest path first (SPF) algorithm lives at the heart of OSPF. The algorithm, developed by Edsger Wybe Dijkstra in 1956, is used by OSPF to provide IP routing with high-speed convergence in a loop-free topology. OSPF provides fast convergence by using triggered, incremental updates that exchange link-state advertisements (LSAs) with neighboring OSPF routers. OSPF is a classless protocol, meaning it carries the subnet mask with all IP routes. It supports a structured two-tiered hierarchical design model using a backbone and other connected areas. This hierarchical design model is used to scale larger networks to further improve convergence time, to create smaller failure domains, and to reduce the complexity of the network routing tables.

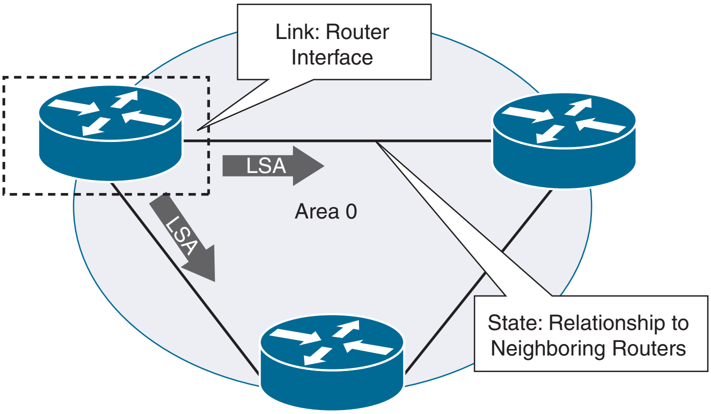
OSPF Characteristics

OSPF is a link-state routing protocol. You can think of a link as an interface on a router. The state of the link is a description of that interface and of its relationship to its neighboring routers. A description of the interface would include, for example, the IP address of the interface, the subnet mask, the type of network to which it is connected, the routers that are connected to that network, and so on. The collection of all these link states forms a link-state database.

OSPF performs the following functions, as illustrated in Figure 25-1:

- Creates a neighbor relationship by exchanging hello packets
- Propagates LSAs rather than routing table updates:
    - **Link:** Router interface
    - **State:** Description of an interface and its relationship to neighboring routers
- Floods LSAs to all OSPF routers in the area, not just the directly connected routers
- Pieces together all the LSAs that OSPF routers generate to create the OSPF link-state database
- Uses the SPF algorithm to calculate the shortest path to each destination and places it in the routing table

Figure 25-1 OSPF Functionality

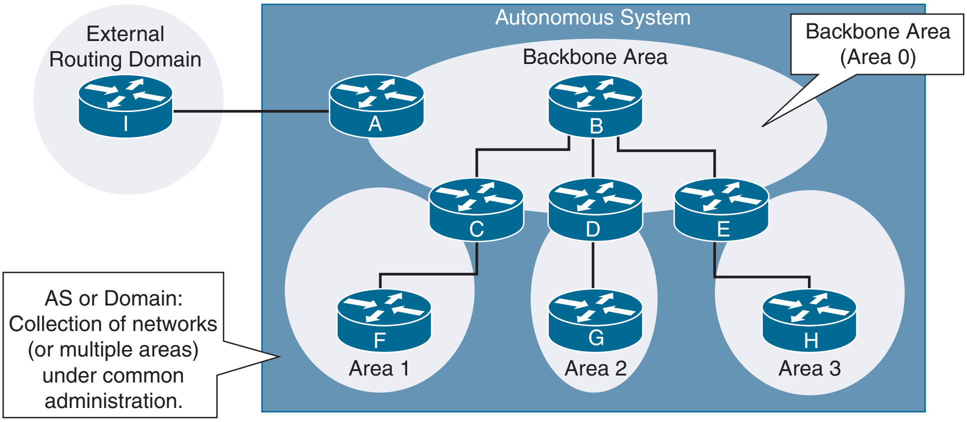


A router sends LSA packets immediately to advertise its state when there are state changes. The router sends the packets periodically as well (every 30 minutes by default). The information about the attached interfaces, the metrics that are used, and other variables are included in OSPF LSAs. As OSPF routers accumulate link-state information, they use the SPF algorithm to calculate the shortest path to each node.

A topological (link-state) database is, essentially, an overall picture of the networks in relationship to the other routers. The topological database contains the collection of LSAs that all routers in the same area have sent. Because the routers in the same area share the same information, they have identical topological databases.

OSPF can operate within a hierarchy. The largest entity in the hierarchy is the autonomous system (AS), which is a collection of networks under a common administration that shares a common routing strategy. An AS can be divided into several areas, which are groups of contiguous networks and attached hosts. Within each AS, a contiguous backbone area must be defined as Area 0. In a multiarea design, all other nonbackbone areas are connected off the backbone area. A multiarea design is effective because the network is segmented to limit the propagation of LSAs inside an area. It is especially useful for large networks. Figure 25-2 illustrates the two-tier hierarchy that OSPF uses in an AS.

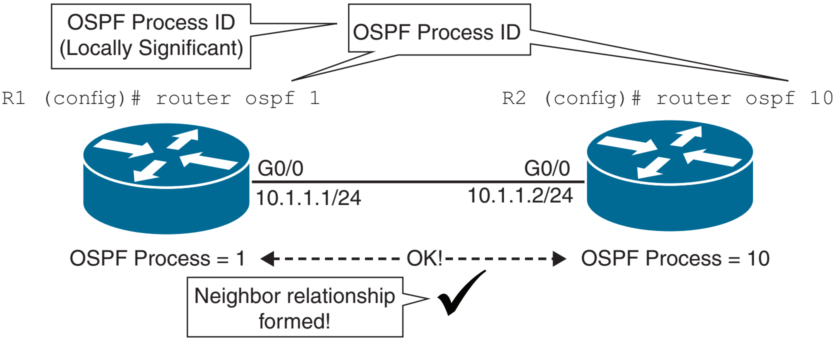
Figure 25-2 OSPF Backbone and Nonbackbone Areas in an AS


External
Routing Domain
I
Autonomous System
Backbone Area
Backbone Area
(Area 0)
A
B
C
D
E
F
G
H
Area 1
Area 2
Area 3
AS or Domain:
Collection of networks
(or multiple areas)
under common
administration.

OSPF Process

Enabling the OSPF process on a device is straightforward. OSPF is started with the same **router ospf** *process-id* command on enterprise routers, multilayer switches, and firewalls. This action requires the configuration of a process ID, which is a value that indicates a unique instance of the OSPF protocol for the device. While this numeric value is needed to start the process, it is not used outside the device on which it is configured, and it is only locally significant (that is, this value is not used for communicating with other OSPF routers). Having one router use OSPF process 10 while a neighboring router uses process 1 will not hinder the establishment of OSPF neighbor relationships. However, for ease of administration, it is best practice to use the same process ID for all devices in the same AS, as shown in Figure 25-3.

Figure 25-3 OSPF Process ID

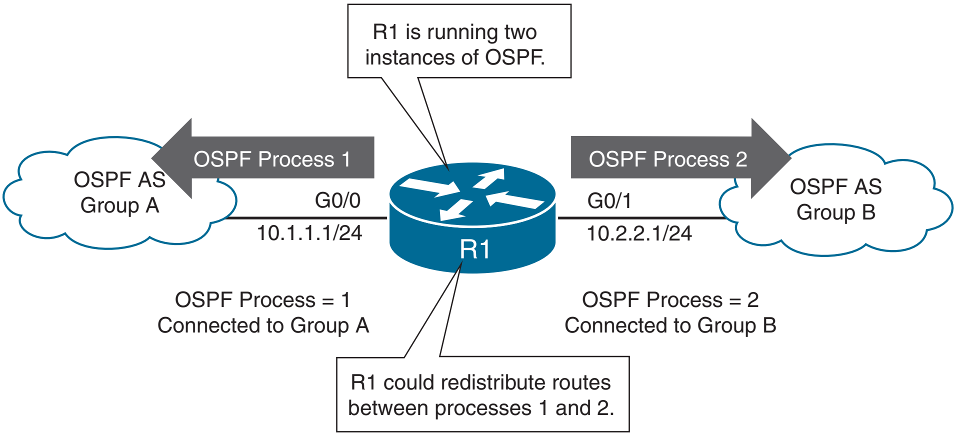


```
R1 (config)# router ospf 1
R2 (config)# router ospf 10
```

OSPF Process ID
(Locally Significant)
OSPF Process ID
G0/0
10.1.1.1/24
G0/0
10.1.1.2/24
OSPF Process = 1
OK!
OSPF Process = 10
Neighbor relationship
formed!

It is possible to have multiple instances of OSPF running on a single router, as illustrated in Figure 25-4. This might be desirable in a situation where two organizations are merging together, and both are running OSPF. The routers designated to merge these two organizations would run an instance of OSPF to communicate to "Group A" and a separate instance for "Group B." The router could redistribute the routing data between the OSPF processes. Another situation in which multiple OSPF processes might be used on a single router is in a service provider's implementation of MPLS. However, it is generally uncommon to need multiple OSPF processes on a router.

Figure 25-4 OSPF Multiple Process IDs

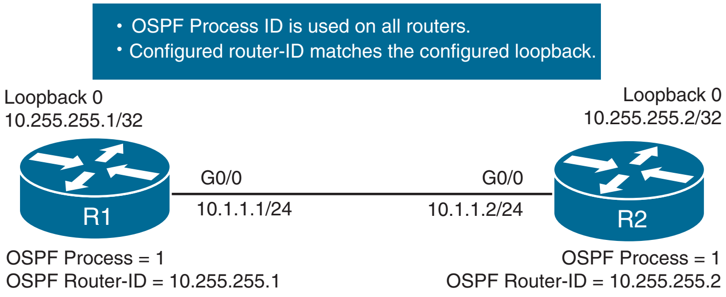


Once the process is started, the OSPF router is assigned a router ID. This ID value is a 32-bit number that is written like an IP address. The ID value is not required to be a valid IP address, but using a valid IP address makes troubleshooting OSPF easier. Whenever the router advertises routes within OSPF, it uses this router ID to mark it as the originator of the routes. Therefore, it is important to ensure that each router within an OSPF network has a unique router ID.

The router ID selection process occurs when the **router ospf** command is entered. Ideally, the command **router-id** *router-id* is used under the OSPF process. If the device does not have an explicit ID assignment, OSPF designates a router ID based on one of the IP addresses (the highest IP address) assigned to the interfaces of the router. If a loopback interface has been created and is active, OSPF uses the IP address of the loopback interface as the router ID. If multiple loopback interfaces are created, OSPF chooses the loopback interface with the numerically highest IP address to use as the router ID. In the absence of loopback interfaces, OPSF chooses an active physical interface with the highest IP address to use for the router ID.

Figure 25-5 shows the configuration of loopback interfaces and the router ID on R1 and R2. The best practice before starting OSPF is to first create a loopback interface and assign it an IP address. Start the OSPF process and then use the **router-id** *router-id* command, entering the IP address of the loopback interface as the router ID.

Figure 25-5 OSPF Router ID Configuration


• OSPF Process ID is used on all routers.
• Configured router-ID matches the configured loopback.
Loopback 0
10.255.255.1/32
Loopback 0
10.255.255.2/32
G0/0
G0/0
10.1.1.1/24
10.1.1.2/24
R1
R2
OSPF Process = 1
OSPF Router-ID = 10.255.255.1
OSPF Process = 1
OSPF Router-ID = 10.255.255.2

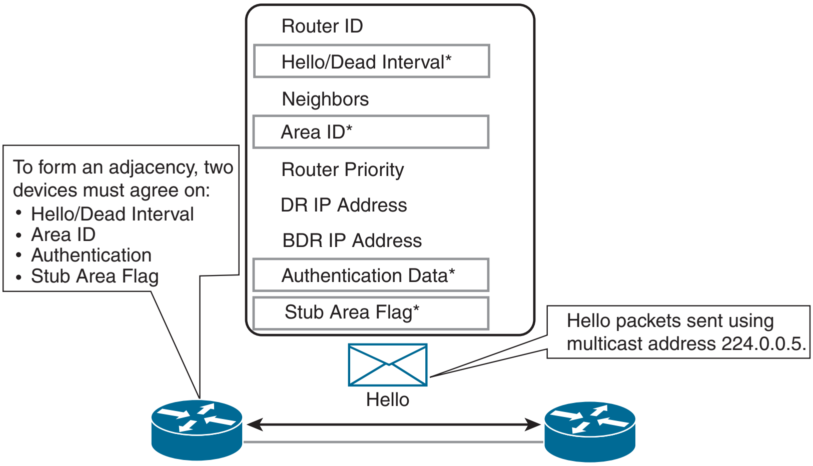
```
R1 (config)# interface loopback 0
R1 (config-if)# ip address 10.255.255.1 255.255.255.255
R1 (config)# router ospf 1
R1 (config-router)# router-id 10.255.255.1
R2 (config)# interface loopback 0
R2 (config-if)# ip address 10.255.255.2 255.255.255.255
R2 (config)# router ospf 1
R1 (config-router)# router-id 10.255.255.2
```

OSPF Neighbor Adjacencies

Neighbor OSPF routers must recognize each other on the network before they can share information because OSPF routing depends on the status of the link between two routers. Hello messages initiate and maintain this process. OSPF routers send hello packets on all OSPF-enabled interfaces to determine whether there are any neighbors on those links.

The Hello protocol establishes and maintains neighbor relationships by ensuring bidirectional (two-way) communication between neighbors.

Figure 25-6 OSPF Hello Message


Router ID
Hello/Dead Interval*
Neighbors
Area ID*
Router Priority
DR IP Address
BDR IP Address
Authentication Data*
Stub Area Flag*
To form an adjacency, two devices must agree on:
• Hello/Dead Interval
• Area ID
• Authentication
• Stub Area Flag
Hello packets sent using multicast address 224.0.0.5.
Hello

Each interface that participates in OSPF uses the multicast address 224.0.0.5 to periodically send hello packets. As shown in Figure 25-6, a hello packet contains the following information:

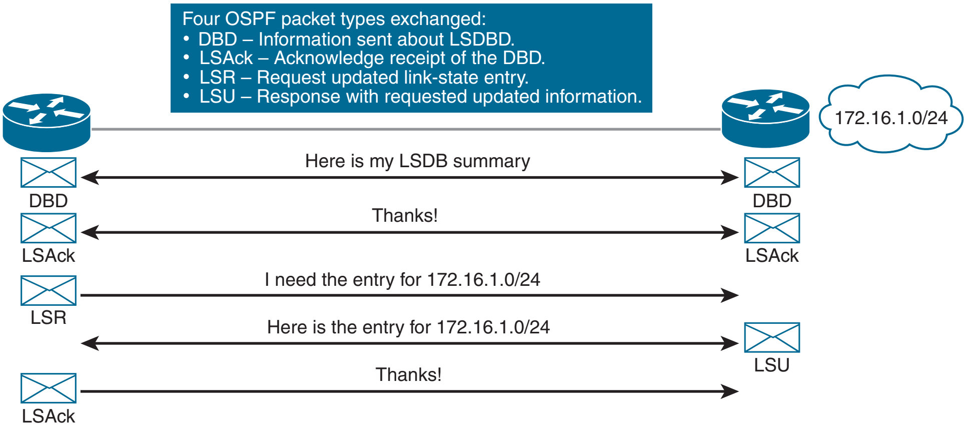
- **Router ID:** The router ID is a 32-bit number that uniquely identifies the router.
- **Hello and dead intervals:** The hello interval specifies the frequency, in seconds, at which a router sends hello packets. The default hello interval on multiaccess networks is 10 seconds. The dead interval is the time, in seconds, that a router waits to hear from a neighbor before declaring the neighboring router out of service. By default, the dead interval is four times the hello interval, or 40 seconds. These timers must be the same on neighboring routers; otherwise, an adjacency is not established.
- **Neighbors:** The Neighbors field lists the adjacent routers with an established bidirectional communication. This bidirectional communication is indicated when the router recognizes itself when it is listed in the Neighbors field of the hello packet from the neighbor.
- **Area ID:** To communicate, two routers must share a common segment, and their interfaces must belong to the same OSPF area on that segment. The neighbors must also share the same subnet and mask. These routers in the same area all have the same link-state information for that area.
- **Router priority:** The router priority is an 8-bit number that indicates the priority of a router. OSPF uses the priority to select a designated router (DR) and a backup designated router (BDR). In certain types of networks, OSPF elects DRs and BDRs. The DR acts as a pseudonode or virtual router to reduce LSA traffic between routers and reduce the number of OSPF adjacencies on the segment.
- **DR and BDR IP addresses:** These addresses are the IP addresses of the DR and BDR for the specific network, if they are known and/or needed, based on the network type.
- **Authentication data:** If router authentication is enabled, two routers must exchange the same authentication data. Authentication is not required, but it is highly recommended. If it is enabled, all peer routers must have the same key configured.
- **Stub area flag:** A stub area is a special area. Designating a stub area is a technique that reduces routing updates by replacing them with a default route. Two routers must also agree on the stub area flag in the hello packets to become neighbors.

OSPF neighbor adjacencies are critical to the operation of OSPF. OSPF proceeds to the phase of exchanging the routing database following the discovery of a neighbor. In other words, without a neighbor relationship, OSPF cannot route traffic. It is important to ensure that the hello/dead timers, area IDs, authentication, and stub area flag information are consistent and match within the hello messages for all devices that intend to establish OSPF neighbor relationships. The neighboring routers must have the same values set for these options.

Building a Link-State Database

When two routers discover each other and establish adjacency by using hello packets, they then exchange information about LSAs. As shown in Figure 25-7, this process operates as follows:

Figure 25-7 OSPF LSDB Sync


Four OSPF packet types exchanged:
• DBD – Information sent about LSDBD.
• LSAck – Acknowledge receipt of the DBD.
• LSR – Request updated link-state entry.
• LSU – Response with requested updated information.
172.16.1.0/24
Here is my LSDB summary
DBD
DBD
Thanks!
LSAck
LSAck
I need the entry for 172.16.1.0/24
LSR
Here is the entry for 172.16.1.0/24
LSU
Thanks!
LSAck

1. The routers exchange one or more DBD (database description or type 2 OSPF) packets. A DBD includes information about the LSA entry header that appears in the link-state database (LSDB) of the router. Each LSA entry header includes information about the link-state type, the address of the advertising router, the cost of the link, and the sequence number. The router uses the sequence number to determine the "newness" of the received link-state information.

2. When the router receives the DBD, it acknowledges the receipt of the DBD that is using the link-state acknowledgment (LSAck) packet.

3. The routers compare the information they receive with the information they have. If the received DBD has a more up-to-date link-state entry, the router sends a link-state request (LSR) to the other router to request the updated link-state entry.

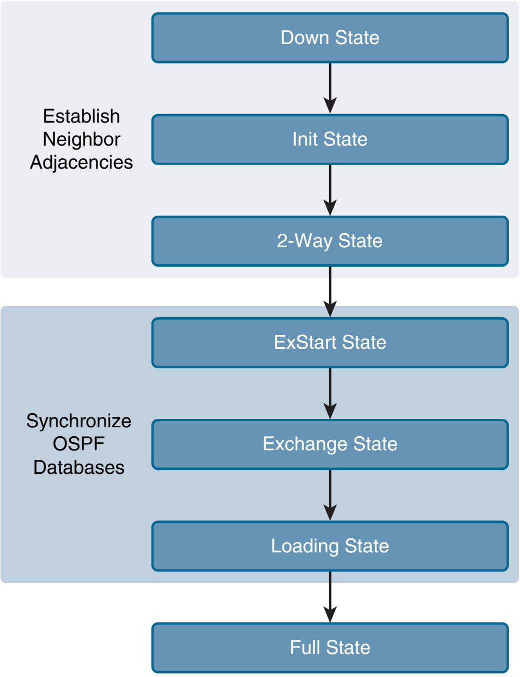
4. The other router responds with complete information about the requested entry in a link-state update (LSU) packet. The LSU contains one or more LSAs. The other router adds the new link-state entries to its LSDB.

5. Finally, when the router receives an LSU, it sends an LSAck.

OSPF Neighbor States

OSPF neighbors go through multiple neighbor states before forming a full OSPF adjacency, as illustrated in Figure 25-8.

Figure 25-8 OSPF Neighbor States



The following is a summary of the states that an interface passes through before establishing an adjacency with another router:

- **Down:** No information has been received on the segment.
- **Init:** The interface has detected a hello packet coming from a neighbor, but bidirectional communication has not yet been established.
- **2-Way:** There is bidirectional communication with a neighbor. The router has seen itself in the hello packets coming from a neighbor. At the end of this stage, the DR and BDR election will be performed if necessary. When routers are in the 2-Way state, they must decide whether to proceed in building an adjacency. The decision is based on whether one of the routers is a DR or BDR or if the link is a point-to-point link or a virtual link.
- **ExStart:** Routers try to establish the initial sequence number that is going to be used in the information exchange packets. The sequence number ensures that routers always get the most recent information. One router becomes the master, and the other becomes the slave. The master router polls the slave for information.
- **Exchange:** Routers describe their entire LSDB by sending database description (DBD) packets. In this state, packets may be flooded to other interfaces on the router.
- **Loading:** In this state, routers finalize the information exchange. Routers have built a link-state request list and a link-state retransmission list. Any information that looks incomplete or

outdated is put on the request list. Any update that is sent is put on the retransmission list until it gets acknowledged.

- **Full:** In this state, adjacency is complete. The neighboring routers are fully adjacent. Adjacent routers have similar LSDBs.

OSPF Packet Types

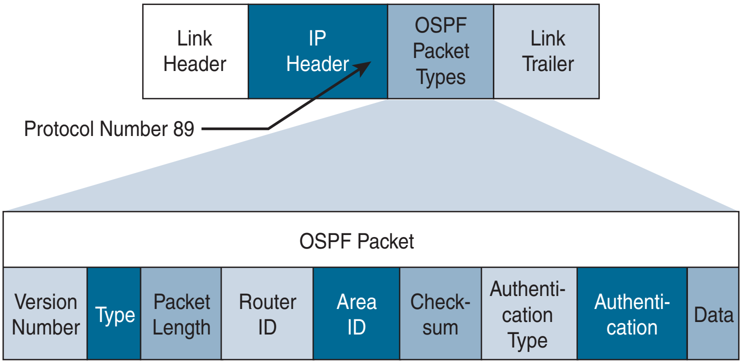
Table 25-1 describes the OSPF packet types.

Table 25-1 OSPF Packet Types

Type	Packet Name	Description
1	Hello	The hello packet discovers and maintains neighbors.
2	DBD	The database description packets contain the LSA headers that help routers build the link-state database.
3	LSR	After DBD packets are exchanged, each router checks the LSA headers against its own database. If it does not have current information for any LSA, it generates an LSR packet and sends it to its neighbor to request updated LSAs.
4	LSU	The LSU packets contain a list of LSAs that should be updated.
5	LSAck	LSAck packets help ensure reliable transmission of LSAs. Each LSA is explicitly acknowledged.

OSPF uses five types of routing protocol packets that share a common protocol header. The Protocol field in the IP header is set to 89. All five packet types are used in normal OSPF operation. All five OSPF packet types are encapsulated directly into an IP payload, as shown in Figure 25-9. OSPF packets do not use TCP or UDP. OSPF requires a reliable packet transport, but because it does not use TCP, OSPF defines an acknowledgment packet (OSPF packet type 5) to ensure reliability.

Figure 25-9 OSPF Packet Encapsulation


Link Header
IP Header
OSPF Packet Types
Link Trailer
Protocol Number 89
OSPF Packet
Version Number
Type
Packet Length
Router ID
Area ID
Check-sum
Authenti-cation Type
Authenti-cation
Data

OSPF LSA Types

Knowing the detailed topology of the OSPF area is a prerequisite for a router to calculate the best paths. Topology details are described by LSAs carried inside LSUs, which are the building blocks of the OSPF LSDB. Individually, LSAs act as database records. In combination, they describe the entire topology of an OSPF network area. Table 25-2 lists the most common LSA types, and the following list describes them:

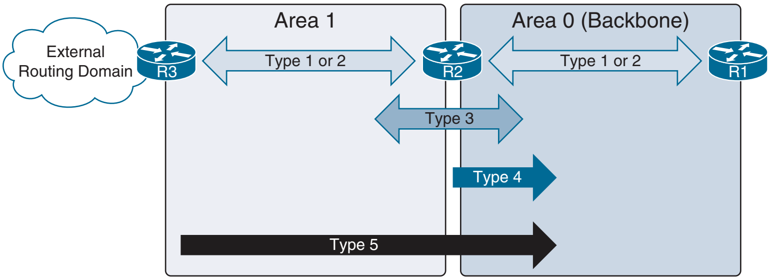
Table 25-2 OSPF LSA Types

LSA Type	Description
1	Router LSA
2	Network LSA
3	Summary LSA
4	ASBR summary LSA
5	AS external LSA
6–11	Other types

- **Type 1:** Every router generates type 1 router LSAs for each area to which it belongs. Router LSAs describe the state of the router links to the area and are flooded only within that particular area. The LSA header contains the link-state ID of the LSA. The link-state ID of the type 1 LSA is the originating router ID.
- **Type 2:** DRs generate type 2 network LSAs for multiaccess networks. Network LSAs describe the set of routers that is attached to a particular multiaccess network. Network LSAs are flooded in the area that contains the network. The link-state ID of the type 2 LSA is the IP interface address of the DR.
- **Type 3:** An ABR takes the information that it learned in one area and describes and summarizes it for another area in the type 3 summary LSA. This summarization is not on by default. The link-state ID of the type 3 LSA is the destination network number.
- **Type 4:** The type 4 ASBR summary LSA tells the rest of the OSPF domain how to get to the ASBR. The link-state ID includes the router ID of the described ASBR.
- **Type 5:** Type 5 AS external LSAs, which are generated by ASBRs, describe routes to destinations that are external to the AS. They get flooded everywhere except into special areas. The link-state ID of the type 5 LSA is the external network number.
- **Type 6:** These specialized LSAs are used in multicast OSPF applications.
- **Type 7:** Type 7 LSAs are used in NSSA special area type for external routes.
- **Type 8 and type 9:** Type 8 and type 9 LSAs are used in OSPFv3 for link-local addresses and intra-area prefixes.
- **Type 10 and type 11:** Type 10 and type 11 LSAs are generic LSAs, also called *opaque*, which allow future extensions of OSPF.

Figure 25-10 shows an example of LSA propagation in which R2 is an ABR between Area 0 and Area 1. R3 acts as the ASBR between the OSPF routing domain and an external domain. LSA types 1 and 2 are flooded between routers within an area. Type 3 and type 5 LSAs are flooded when exchanging information between the backbone and standard areas. Type 4 LSAs are injected into the backbone by the ABR because all routers in the OSPF domain need to reach the ASBR (R3).

Figure 25-10 OSPF LSA Propagation

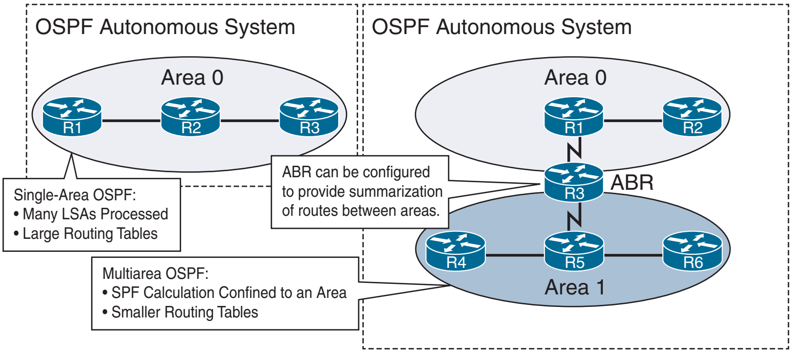

External Routing Domain
Area 1
Area 0 (Backbone)
R3
R2
R1
Type 1 or 2
Type 1 or 2
Type 3
Type 4
Type 5

Single-Area and Multiarea OSPF

The single-area OSPF design has all routers in a single OSPF area. This design results in many LSAs being processed on every router and in larger routing tables. This OSPF configuration follows a single-area design in which all the routers are treated as being internal routers to the area, and all the interfaces are members of this single area.

Keep in mind that OSPF uses flooding to exchange link-state updates between routers. Any change in the routing information is flooded to all routers in an area. For this reason, the single-area OSPF design can become undesirable as the network grows. The number of LSAs that are processed on every router increases, and the routing tables may grow very large.

Figure 25-11 Single-Area and Multiarea OSPF

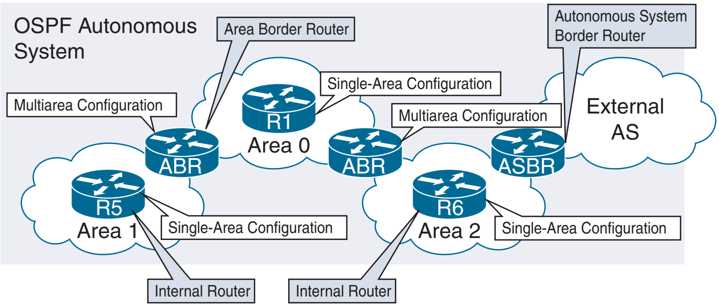

OSPF Autonomous System
Area 0
R1
R2
R3
OSPF Autonomous System
Area 0
R1
R2
R3
ABR
R4
R5
R6
Area 1
Single-Area OSPF:
• Many LSAs Processed
• Large Routing Tables
ABR can be configured to provide summarization of routes between areas.
Multiarea OSPF:
• SPF Calculation Confined to an Area
• Smaller Routing Tables

For enterprise networks, a multiarea design is a better solution than a single-area design. In a multi-area design, the network is segmented to limit the propagation of LSAs inside an area and to make the routing tables smaller by utilizing summarization. In Figure 25-11, an area border router (ABR) is configured between two areas (Area 0 and Area 1). The ABR can provide summarization of routes between the two areas and can act as a default gateway for all Area 1 internal routers (R4, R5, and R6).

There are two types of routers from a configuration point of view, as illustrated in Figure 25-12:

- **Routers with single-area configuration:** Internal routers (R5, R6), the backbone router (R1), and autonomous system border routers (ASBRs) that reside in one area.
- **Routers with a multiarea configuration:** Area border routers (ABRs) and ASBRs that reside in more than one area.

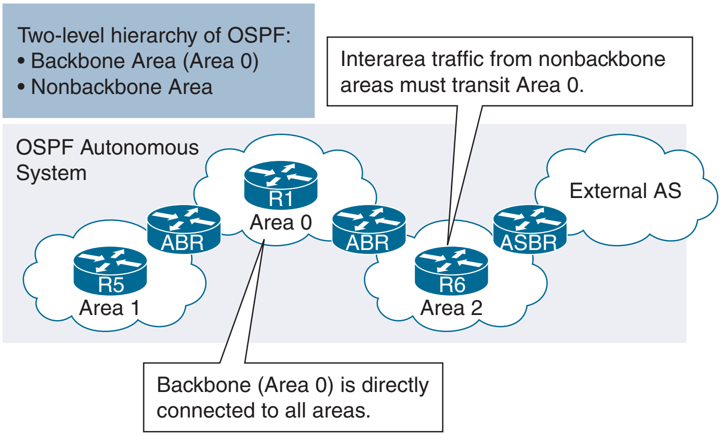
Figure 25-12 OSPF Router Roles


OSPF Autonomous System
Area Border Router
Autonomous System Border Router
Single-Area Configuration
Multiarea Configuration
Multiarea Configuration
R1
Area 0
ABR
ABR
ASBR
External AS
R5
Area 1
R6
Area 2
Single-Area Configuration
Single-Area Configuration
Internal Router
Internal Router

OSPF Area Structure

As mentioned earlier, OSPF uses a two-tiered area hierarchy. Figure 25-13 illustrates the two areas in this hierarchy:

Figure 25-13 OSPF Hierarchy


Two-level hierarchy of OSPF:
• Backbone Area (Area 0)
• Nonbackbone Area
Interarea traffic from nonbackbone areas must transit Area 0.
OSPF Autonomous System
R1
Area 0
ABR
ABR
ASBR
External AS
R5
Area 1
R6
Area 2
Backbone (Area 0) is directly connected to all areas.

- **Backbone area (Area 0):** The primary function of this OSPF area is to quickly and efficiently move IP packets. Backbone areas interconnect with other OSPF area types. The OSPF hierarchical area structure requires that all areas connect directly to the backbone area. Interarea traffic must traverse the backbone.
- **Normal, or nonbackbone, area:** The primary function of this OSPF area is to connect users and resources. Normal areas are usually set up according to functional or geographic groupings. By default, a normal area does not allow traffic from another area to use its links to reach other areas. All interarea traffic from other areas must cross a transit area such as Area 0.

All OSPF areas and routers that are running the OSPF routing protocol compose the OSPF AS.

The routers that are configured in Area 0 are known as *backbone routers*. If a router has any interface(s) in Area 0, it is considered a backbone router. Routers that have all their interfaces in a single area are called *internal routers* because they have to manage only a single LSDB each.

An ABR connects multiple areas together. Normally, this configuration is used to connect Area 0 to the nonbackbone areas. An OSPF ABR plays a very important role in the network design and has interfaces in more than one area. An ABR has the following characteristics:

- It separates LSA flooding zones.
- It becomes the primary point for area address summarization.
- It can designate a nonbackbone area to be a special area type, such as a stub area.
- It maintains the LSDB for each area with which it is connected.

An ASBR connects any OSPF area to a different routing domain. The ASBR is the point where external routes can be introduced into the OSPF AS. Essentially, routers act as an ASBR if routes are introduced into the AS using route redistribution or if the OSPF router is originating the default route. ASBR routers can live in the backbone area or in the nonbackbone area. A device running OSPF can act as an ASBR and as an ABR concurrently.

OSPF Network Types

OSPF defines distinct types of networks, based on the physical link types. OSPF operation is different in each type of network, including how adjacencies are established and which configuration is required. Table 25-3 summarizes the characteristics of the OSPF network types.

Table 25-3 OSPF Network Types

OSPF Network Type	Uses DR/BDR	Default Hello/ Dead Interval (seconds)	Dynamic Neighbor Discovery	More Than Two Routers Allowed in Subnet?
Point-to-point	No	10/40	Yes	No
Broadcast	Yes	10/40	Yes	Yes
Nonbroadcast	Yes	30/120	No	Yes

OSPF Network Type	Uses DR/BDR	Default Hello/ Dead Interval (seconds)	Dynamic Neighbor Discovery	More Than Two Routers Allowed in Subnet?
Point-to-multipoint	No	30/120	Yes	Yes
Point-to-multipoint nonbroadcast	No	30/120	No	Yes
Loopback	No	—	—	No

The following are the network types most commonly defined by OSPF:

- **Point-to-point:** Routers use multicast to dynamically discover neighbors. There is no DR/BDR election because only two routers can be connected on a single point-to-point segment. This is a default OSPF network type for serial links and point-to-point Frame Relay subinterfaces.
- **Broadcast:** Multicast is used to dynamically discover neighbors. The DR and BDR are elected to optimize the exchange of information. This is a default OSPF network type for multiaccess Ethernet links.
- **Nonbroadcast:** This network type is used on networks that interconnect more than two routers but without broadcast capability. Frame Relay and Asynchronous Transfer Mode (ATM) are examples of nonbroadcast multiaccess (NBMA) networks. Neighbors must be statically configured, and then DR/BDR election occurs. This network type is the default for all physical interfaces and multipoint subinterfaces using Frame Relay encapsulation.
- **Point-to-multipoint:** OSPF treats this network type as a logical collection of point-to-point links, although all interfaces belong to the common IP subnet. Every interface IP address appears in the routing table of the neighbors as a host /32 route. Neighbors are discovered dynamically using multicast. There is no DR/BDR election.
- **Point-to-multipoint nonbroadcast:** This network type is a Cisco extension that has the same characteristics as point-to-multipoint, except that neighbors are not discovered dynamically. Neighbors must be statically defined, and unicast is used for communication. This network type can be useful in point-to-multipoint scenarios where multicast and broadcasts are not supported.
- **Loopback:** This is the default network type on loopback interfaces.

OSPF DR and BDR Election

Multiaccess networks, either broadcast (such as Ethernet) or nonbroadcast (such as Frame Relay), present interesting issues for OSPF. All routers sharing the common segment are part of the same IP subnet. When forming adjacency on a multiaccess network, every router tries to establish full OSPF adjacency with all other routers on the segment. This behavior may not be an issue for smaller multiaccess broadcast networks, but it may be an issue for the NBMA, where, usually, you do not have a full-mesh PVC topology. This issue in NBMA networks manifests in the inability for neighbors to synchronize their OSPF databases directly among themselves. A logical solution, in this case, is to have a central point of OSPF adjacency responsible for the database synchronization and advertisement of the segment to the other routers.

As the number of routers on the segment grows, the number of OSPF adjacencies increases exponentially. Every router must synchronize its OSPF database with every other router, and if there are many routers on a segment, this behavior leads to inefficiency. Another issue arises when every router on the segment advertises all its adjacencies to other routers in the network. If you have full-mesh OSPF adjacencies, the other OSPF routers receive a large amount of redundant link-state information. The solution for this problem is again to establish a central point with which every other router forms an adjacency and advertises the segment to the rest of the network.

The routers on the multiaccess segment elect a DR and a BDR that centralize communication for all routers connected to the segment. The DR and BDR improve network functionality in the following ways:

- **Reducing routing update traffic:** The DR and BDR act as a central point of contact for link-state information exchange on a multiaccess network. Therefore, each router must establish a full adjacency with the DR and the BDR. Each router, rather than exchanging link-state information with every other router on the segment, sends the link-state information to the DR and BDR only by using the dedicated multicast address 224.0.0.6. The DR represents the multiaccess network in the sense that it sends link-state information from each router to all other routers in the network. This flooding process significantly reduces the router-related traffic on the segment.

- **Managing link-state synchronization:** The DR and BDR ensure that the other routers on the network have the same link-state information about the common segment. In this way, the DR and BDR reduce the number of routing errors.

When the DR is operating, the BDR does not perform any DR functions. Instead, the BDR receives all the information, but the DR performs the LSA forwarding and LSDB synchronization tasks. The BDR performs the DR tasks only if the DR fails. When the DR fails, the BDR automatically becomes the new DR, and a new BDR election occurs.

When routers start establishing OSPF neighbor adjacencies, they first send OSPF hello packets to discover which OSPF neighbors are active on the common Ethernet segment. After the bidirectional communication between routers is established and they are all in OSPF neighbor 2-Way state, the DR/BDR election process begins.

One of the fields in the OSPF hello packet that is used in the DR/BDR election process is the Router Priority field. Every broadcast and nonbroadcast multiaccess OSPF-enabled interface has an assigned priority value, which is a number between 0 and 255. By default, in Cisco IOS Software, the OSPF interface priority value is 1. You can manually change it using the **ip ospf priority** interface-level command. To elect a DR and BDR, the routers view the OSPF priority value of other routers during the hello packet exchange process and then use the following conditions to determine which router to select:

- The router with the highest priority value is elected as the DR.

- The router with the second-highest priority value is the BDR.

- If there is a tie, where two routers have the same priority value, the router ID is used as the tiebreaker. The router with the highest router ID becomes the DR. The router with the second-highest router ID becomes the BDR.

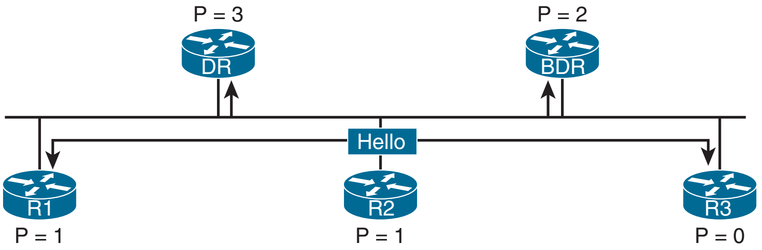
- A router with a priority that is set to 0 cannot become the DR or BDR. A router that is not the DR or BDR is called a DROTHER.

The DR/BDR election process takes place on broadcast and nonbroadcast multiaccess networks. The main difference between the two is the type of IP address that is used in the hello packet. On multiaccess broadcast networks, routers use multicast destination IP address 224.0.0.6 to communicate with the DR (called AllDRRouters), and the DR uses multicast destination IP address 224.0.0.5 to communicate with all other non-DR routers (called AllSPFRouters). On NBMA networks, the DR and adjacent routers communicate using unicast.

The DR/BDR election procedure occurs not only when the network first becomes active, but it also occurs when the DR becomes unavailable. In this case, the BDR immediately becomes the DR, and the election of the new BDR starts.

Figure 25-14 illustrates the OSPF DR and BDR election process. The router with a priority of 3 is chosen as DR, and the router with a priority of 2 is chosen as BDR. Notice that R3 has a priority value of 0. This places it in a permanent DROTHER state.

Figure 25-14 OSPF DR and BDR Election



OSPF Timers

Like EIGRP, OSPF uses two timers to check neighbor reachability. These two timers are named the hello and dead timers. The values of the hello and dead intervals are carried in the OSPF hello packet, which serves as a keepalive message that acknowledges the router's presence on the segment. The hello interval specifies the frequency at which OSPF hello packets are sent, in seconds. The SPF dead timer specifies how long a router waits to receive a hello packet before it declares the neighbor router down.

OSPF requires that both the hello and dead timers be identical for all routers on the segment to become OSPF neighbors. The default value of the OSPF hello timer on multiaccess broadcast and point-to-point links is 10 seconds, and on all other network types, including NBMA, it is 30 seconds. Once you set up the hello interval, the default value of the dead interval is automatically four times the hello interval. For broadcast and point-to-point links, it is 40 seconds, and for all other OSPF network types, it is 120 seconds.

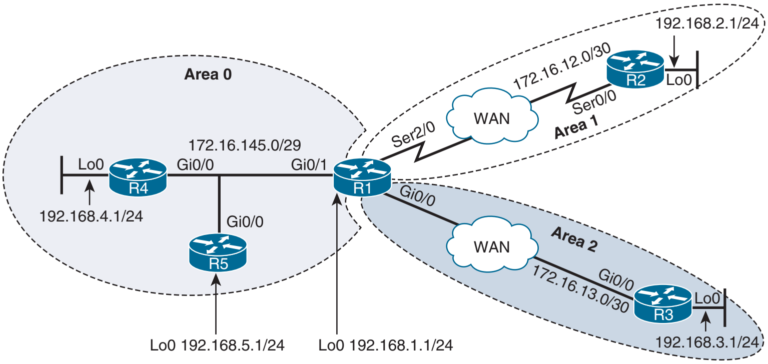
To detect topological changes more quickly, you can lower the value of the OSPF hello interval; the downside is more routing traffic on the link.

The OSPF timers can be changed by using the **ip ospf hello-interval** and **ip ospf dead-interval interface configuration** commands.

Multiarea OSPF Configuration

Figure 25-15 illustrates the topology used for the multiarea OSPF example that follows. R1, R4, and R5 are connected to a common multiaccess Ethernet segment. R1 and R2 are connected over a point-to-point serial link. R1 and R3 are connected over an Ethernet WAN link. All routers are configured with the correct physical and logical interfaces and IP addresses. The OSPF router ID is configured to match the individual router's Loopback 0 interface. Example 25-1 shows the basic multiarea OSPF configuration for all five routers.

Figure 25-15 Multiarea OSPF Basic Configuration Example


Area 0
Lo0
R4
192.168.4.1/24
172.16.145.0/29
Gi0/0
Gi0/1
Gi0/0
R5
Lo0 192.168.5.1/24
R1
Lo0 192.168.1.1/24
Ser2/0
WAN
172.16.12.0/30
Ser0/0
Area 1
R2
Lo0
192.168.2.1/24
Gi0/0
WAN
Area 2
172.16.13.0/30
Gi0/0
R3
Lo0
192.168.3.1/24

Example 25-1 Configuring Multiarea OSPF

```
R1(config)# router ospf 1
R1(config-router)# network 192.168.1.0.0 0.0.0.255 area 0
R1(config-router)# network 172.16.145.0 0.0.0.7 area 0
R1(config-router)# network 172.16.12.0 0.0.0.3 area 1
R1(config-router)# network 172.16.13.0 0.0.0.3 area 2
R1(config-router)# router-id 192.168.1.1
```

```
R2(config)# router ospf 1
R2(config-router)# network 172.16.12.0 0.0.0.3 area 1
R2(config-router)# network 192.168.2.0 0.0.0.255 area 1
R1(config-router)# router-id 192.168.2.1
```

```
R3(config)# router ospf 1
R3(config-router)# network 172.16.13.2 0.0.0.0 area 2
R3(config-router)# interface Loopback 0
R3(config-if)# ip ospf 1 area 2
R1(config-router)# router-id 192.168.3.1
```

```
R4(config)# router ospf 1
R4(config-router)# network 172.16.145.0 0.0.0.7 area 0
R4(config-router)# network 192.168.4.0 0.0.0.255 area 0
R4(config-router)# router-id 192.168.4.1
```

```
R5(config)# router ospf 1
R5(config-router)# network 172.16.145.0 0.0.0.7 area 0
R5(config-router)# network 192.168.5.0 0.0.0.255 area 0
R5(config-router)# router-id 192.168.5.1
```

To enable the OSPF process on the router, use the **router ospf** *process-id* command.

There are multiple ways to enable OSPF on an interface. To define interfaces on which OSPF process runs and to define the area ID for those interfaces, use the **networ**k *ip-address wildcard-mask* **area** *area-id* command. The combination of *ip-address* and *wildcard-mask* allows you to define one or multiple interfaces to be associated with a specific OSPF area using a single command.

Notice on R3 the use of the 0.0.0.0 wildcard mask with the **network** command. This mask indicates that only the interface with the specific IP address listed is enabled for OSPF.

Another method exists for enabling OSPF on an interface. R3's Loopback 0 interface is included in Area 2 by using the **ip ospf** *process-id* **area** *area-id* command. This method explicitly adds the interface to Area 2 without the use of the **network** command. This capability simplifies the configuration of unnumbered interfaces with different areas and ensures that any new interfaces brought online would not automatically be included in the routing process. This configuration method is also used for OSPFv3 since that routing protocol doesn't allow the use of the **network** statement.

The **router-id** command is used on each router to hard code the Loopback 0 IP address as the OSPF router ID.

Verifying OSPF Functionality

You can use the following **show** commands to verify how OSPF is behaving:

- **show ip ospf interface [brief]**
- **show ip ospf neighbor**
- **show ip route ospf**

Example 25-2 shows these commands applied to the previous configuration example.

Example 25-2 Verifying Multiarea OSPF

```
R1# show ip ospf interface
Loopback0 is up, line protocol is up
  Internet Address 192.168.1.1/24, Area 0, Attached via Network Statement
  Process ID 1, Router ID 192.168.1.1, Network Type LOOPBACK, Cost: 1
  Topology-MTID    Cost    Disabled    Shutdown      Topology Name
        0           1         no          no            Base
  Loopback interface is treated as a stub Host
GigabitEthernet0/1 is up, line protocol is up
  Internet Address 172.16.145.1/29, Area 0, Attached via Network Statement
  Process ID 1, Router ID 192.168.1.1, Network Type BROADCAST, Cost: 10
  Topology-MTID    Cost    Disabled    Shutdown      Topology Name
        0           10        no          no            Base
  Transmit Delay is 1 sec, State DROTHER, Priority 1
  Designated Router (ID) 192.168.5.1, Interface address 172.16.145.5
  Backup Designated router (ID) 192.168.4.1, Interface address 172.16.145.4
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
    Hello due in 00:00:05
<. . . output omitted . . .>
Serial2/0 is up, line protocol is up
  Internet Address 172.16.12.1/30, Area 1, Attached via Network Statement
  Process ID 1, Router ID 192.168.1.1, Network Type POINT_TO_POINT, Cost: 64
<. . . output omitted . . .>
GigabitEthernet0/0 is up, line protocol is up
  Internet Address 172.16.13.1/30, Area 2, Attached via Network Statement
  Process ID 1, Router ID 192.168.1.1, Network Type BROADCAST, Cost: 10
<. . . output omitted . . .>
```

```
R1# show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Lo0          1     0               192.168.1.1/24     1     LOOP  0/0
Gi0/1        1     0               172.16.145.1/29    1     DROTH 2/2
Se2/0        1     1               172.16.12.1/30     64    P2P   1/1
Gi0/0        1     2               172.16.13.1/30     1     BDR   1/1
```

```
R1# show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
192.168.4.1       1   FULL/BDR        00:00:33    172.16.145.4    GigabitEthernet0/1
```

```
192.168.5.1      1    FULL/DR        00:00:36    172.16.145.5    GigabitEthernet0/1
192.168.2.1      1    FULL/ -        00:01:53    172.16.12.2     Serial2/0
192.168.3.1      1    FULL/DR        00:00:36    172.16.13.2     GigabitEthernet0/0
```

```
R4# show ip route ospf
Codes:  L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
        D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
        N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
        E1 - OSPF external type 1, E2 - OSPF external type 2
        i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
        ia - IS-IS inter area, * - candidate default, U - per-user static route
        o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
        + - replicated route, % - next hop override

Gateway of last resort is not set

      172.16.0.0/16 is variably subnetted, 4 subnets, 3 masks
O IA     172.16.12.0/30 [110/74] via 172.16.145.1, 00:34:57, Ethernet0/0
O IA     172.16.13.0/30 [110/20] via 172.16.145.1, 00:36:17, Ethernet0/0
      192.168.1.0/32 is subnetted, 1 subnets
O        192.168.1.1 [110/11] via 172.16.145.1, 00:36:58, Ethernet0/0
      192.168.2.0/32 is subnetted, 1 subnets
O IA     192.168.2.1 [110/75] via 172.16.145.1, 00:34:57, Ethernet0/0
      192.168.3.0/32 is subnetted, 1 subnets
O IA     192.168.3.1 [110/21] via 172.16.145.1, 00:36:17, Ethernet0/0
      192.168.5.0/32 is subnetted, 1 subnets
O        192.168.5.1 [110/11] via 172.16.145.5, 01:12:29, Ethernet0/0
```

In Example 25-2, the **show ip ospf interface** command lists all the OSPF-enabled interfaces on R1. The output includes the IP address, the area the interface is in, the OSPF network type, the OSPF state, the DR and BDR router IDs (if applicable), and the OSPF timers. The **show ip ospf interface brief** command provides similar but simpler output. The **show ip ospf neighbor** command lists the router's OSPF neighbors as well as their router ID, interface priority, OSPF state, dead time, IP address, and the interface used by the local router to reach the neighbor.

The **show ip route ospf** command is executed on router R4. Among routes that are originated within an OSPF autonomous system, OSPF clearly distinguishes two types of routes: intra-area routes and interarea routes. Intra-area routes are routes that are originated and learned in the same local area. The character O is the code for the intra-area routes in the routing table. The second type is interarea routes, which originate in other areas and are inserted into the local area to which your router belongs. The characters O IA are the code for the interarea routes in the routing table. Interarea routes are inserted into other areas by the ABR.

The prefix 192.168.5.0/32 is an example of an intra-area route from the perspective of R4. It originated from router R5, which is part of Area 0, the same area in which R4 resides.

The prefixes from R2 and R3, which are part of Area 1 and Area 2, respectively, are shown in the routing table on R4 as interarea routes. The prefixes were inserted into Area 0 as interarea routes by R1, which plays the role of ABR.

The prefixes for all router loopbacks (192.168.1.0/24, 192.168.2.0/24, 192.168.3.0/24, 192.168.5.0/24) are displayed in the R4 routing table as host routes 192.168.1.1/32, 192.168.2.1/32, 192.168.3.1/32, and 192.168.5.1/32. By default, OSPF advertises any subnet that is configured on a loopback interface as a /32 host route. To change this default behavior, you can change the OSPF network type on the loopback interface from the default **loopback** to **point-to-point** by using the **ip ospf network point-to-point** interface configuration command.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	8, 9
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	5

This page intentionally left blank

Day 24

Advanced OSPFv2 and OSPFv3

ENCOR 350-401 Exam Topics

Infrastructure

- Layer 3
    - Configure and verify simple OSPF environments, including multiple normal areas, summarization, and filtering (neighbor adjacency, point-to-point and broadcast network types, and passive interface)

Key Topics

Today we review advanced OSPFv2 optimization features, such as OSPF cost manipulation, route filtering, summarization, and default routing. We also look at OSPFv3 configuration and tuning using the newer address family (AF) framework that supports IPv4 and IPv6.

OSPF Cost

A metric is an indication of the overhead required to send packets across a certain interface. OSPF uses cost as a metric. A smaller cost indicates a better path than a higher cost. By default, on Cisco devices, the cost of an interface is inversely proportional to the bandwidth of the interface, so a higher bandwidth has a lower OSPF cost since it takes longer for packets to cross a 10 Mbps link than a 1 Gbps link.

The formula that you use to calculate OSPF cost is:

$$Interface\ Cost = \frac{Reference\ Bandwidth}{Interface\ Bandwidth\ (bps)}$$

The default reference bandwidth is 10^8, which is 100,000,000. This is equivalent to the bandwidth of a Fast Ethernet interface. Therefore, the default cost of a 10 Mbps Ethernet link is $10^8 / 10^7 = 10$, and the cost of a 100 Mbps link is $10^8 / 10^8 = 1$.

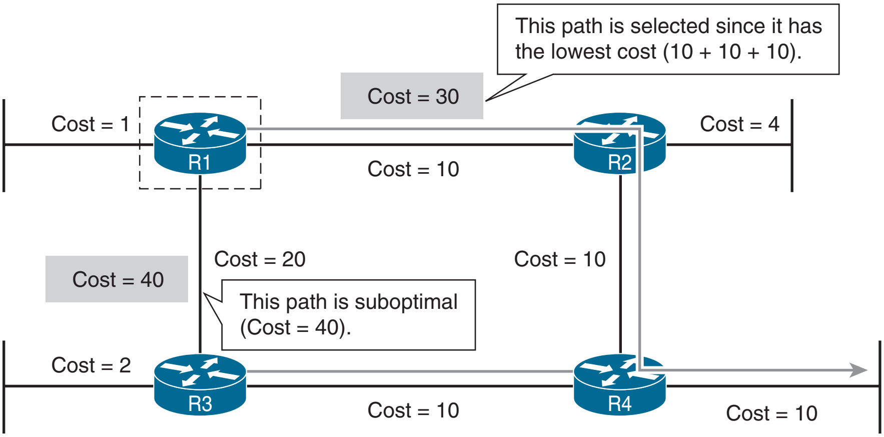
A problem arises with links that are faster than 10 Mbps. Because the OSPF cost has to be a positive integer, all links that are faster than Fast Ethernet have an OSPF cost of 1. Since most networks today are operating with faster speeds, it may be a good idea to consider changing the default reference bandwidth value on all routers within an AS. However, you need to be aware of the consequences of making such changes. Because the link cost is a 16-bit number, increasing the reference bandwidth to differentiate between high-speed links might result in a loss of differentiation in your low-speed links. The 16-bit value provides OSPF with a maximum cost value of 65,535 for a single link. If the reference bandwidth were changed to 10^{11}, 100 Gpbs links would have a value of 1, 10 Gpbs links would be 10, and so on. The issue is that for a T1 link, the cost is now 64,766 (10^{11}/1.544 Mbps) and anything slower than that will now have the largest OSPF cost value of 65,535.

To improve OSPF behavior, you can adjust the reference bandwidth to a higher value by using the **auto-cost reference-bandwidth** OSPF configuration command. Note that this setting is local to each router. If this setting is used, it is recommended that it be applied consistently across the network. You can indirectly set the OSPF cost by configuring the **bandwidth** *speed* interface subcommand (where *speed* is in Kbps). In such cases, the formula shown earlier is used—just with the configured bandwidth value. The most controllable method of configuring OSPF costs, but the most laborious, is to configure the interface cost directly. Using the **ip ospf cost** interface configuration command, you can directly change the OSPF cost of a specific interface. The cost of the interface can be set to a value between 1 and 65,535. This command overrides whatever value is calculated based on the reference bandwidth and the interface bandwidth.

Shortest Path First Algorithm

The shortest path first (SPF), or Dijkstra, algorithm places each router at the root of the OSPF tree and then calculates the shortest path to each node. The path calculation is based on the cumulative cost that is required to reach that destination. For example, in Figure 24-1, R1 has calculated a total cost of 30 to reach the R4 LAN via R2 and a total of 40 to reach the same LAN via R3. The path with a cost of 30 will be chosen as the best path in this case because a lower cost is better.

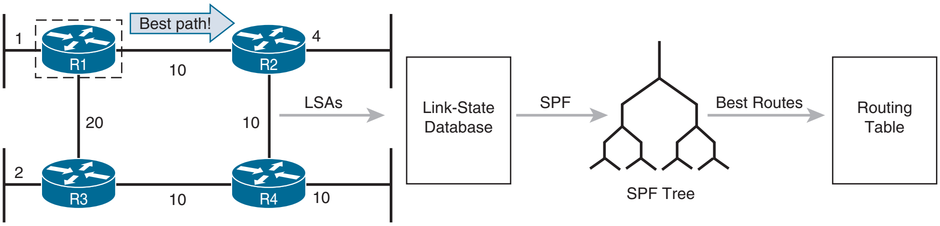
Figure 24-1 OSPF Cost Calculation Example


This path is selected since it has the lowest cost (10 + 10 + 10).
Cost = 30
Cost = 1
R1
Cost = 10
R2
Cost = 4
Cost = 20
Cost = 10
Cost = 40
This path is suboptimal (Cost = 40).
Cost = 2
R3
Cost = 10
R4
Cost = 10

Link-state advertisements (LSAs) are flooded throughout the area using a reliable process, which ensures that all the routers in an area have the same topological database. Each router uses the information in its topological database to calculate a shortest path tree, with itself as the root. The router then uses this tree to route network traffic.

Figure 24-2 shows the R1 view of the network, where R1 is the root and calculates the pathways to every other device based on itself as the root. Keep in mind that each router has its own view of the topology, even though all the routers build the shortest path trees by using the same link-state database.

Figure 24-2 OSPF SPF Tree



R1 SPF Tree

Destination	Shortest Path	Cost
R2 LAN	R1 to R2	14
R3 LAN	R1 to R3	22
R4 LAN	R1 to R2 to R4	30

LSAs are flooded through the area in a reliable manner with OSPF, which ensures that all routers in an area have the same topological database. Because of the flooding process, R1 has learned the link-state information for each router in its routing area. Each router uses the information in its topological database to calculate a shortest path tree with itself as the root. The tree is then used to populate the IP routing table with the best paths to each network.

For R1, the shortest path to each LAN and its cost are shown in Figure 24-2. The shortest path is not necessarily the best path. Each router has its own view of the topology, even though the routers build shortest path trees by using the same link-state database. Unlike with EIGRP, when OSPF determines the shortest path based on all possible paths, it discards any information pertaining to these alternate paths. Any paths not marked as "shortest" would be trimmed from the SPF tree list. During a topology change, the Dijkstra algorithm is run to recalculate the shortest paths for any affected subnets.

OSPF Passive Interfaces

Passive interface configuration is a common method for hardening routing protocols and reducing the use of resources. It is also supported by OSPF.

Use the **passive-interface default** router configuration command to enable this feature for all interfaces or use the **passive-interface** *interface-id* router configuration command to make specific interfaces passive.

When you configure a passive interface under the OSPF process, the router stops sending and receiving OSPF hello packets on the selected interface. Use passive interface configuration only on interfaces where you do not expect the router to form any OSPF neighbor adjacency. When you use the passive interface setting as default, you can identify interfaces that should remain active with the **no passive-interface** configuration command.

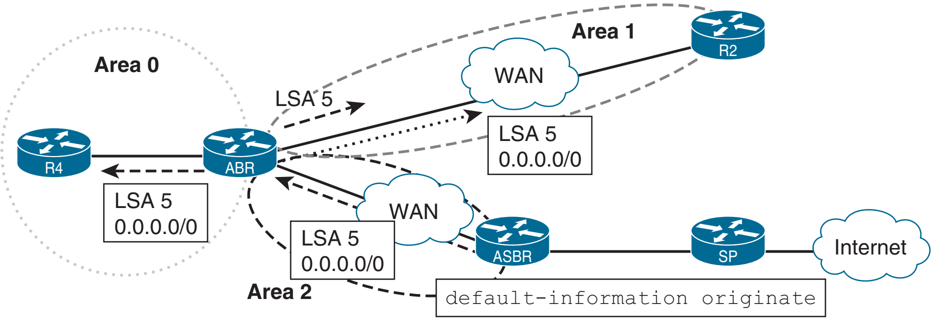
OSPF Default Routing

To be able to perform routing from an OSPF domain toward external networks or toward the Internet, you must either know all the destination networks or create a default route noted as 0.0.0.0/0.

The default routes provide the most scalable approach. Default routing guarantees smaller routing tables and ensures that fewer resources are consumed on the routers. There is no need to recalculate the SPF algorithm if one or more networks fail.

To implementing default routing in OSPF, you can inject a default route using a type 5 AS external LSA. You implement this by using the **default-information originate** command on the uplink ASBR, as shown in Figure 24-3. The uplink ASBR connects the OSPF domain to the upstream router in the SP network. The uplink ASBR generates a default route using a type 5 AS external LSA, which is flooded in all OSPF areas except the stub areas.

Figure 24-3 OSPF Default Routing



You can use different keywords in the configuration command. To advertise 0.0.0.0/0 regardless of whether the advertising router already has a default route in its own routing table, add the keyword **always** to the **default-information originate** command, as shown in this example:

```
ASBR(config-router)# default-information originate ?
  always        Always advertise default route
  metric        OSPF default metric
  metric-type   OSPF metric type for default routes
  route-map     Route-map reference
  <cr>
```

The router participating in an OSPF network automatically becomes an ASBR when you use the **default-information originate** command. You can also use a route map to define dependency on any condition inside the route map. The **metric** and **metric-type** options allow you to specify the OSPF cost and metric type of the injected default route.

After you configure the ASBR to advertise a default route into OSPF, all other routers in the topology should receive it. Example 24-1 shows the routing table on R4 from Figure 24-3. Notice that R4 lists the default route as an O* E2 route in the routing table because it is learned through a type 5 AS external LSA.

Example 24-1 Verifying the Routing Table on R4

```
R4# show ip route ospf
<. . . output omitted . . .>

Gateway of last resort is 172.16.25.2 to network 0.0.0.0

O*E2  0.0.0.0/0 [110/1] via 172.16.25.2, 00:13:28, GigabitEthernet0/0
<. . . output omitted . . .>
```

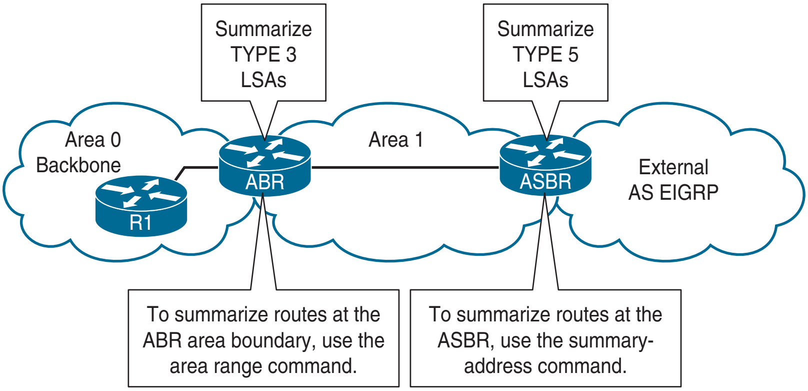
OSPF Route Summarization

In large internetworks, hundreds or even thousands of network addresses can exist. It is often problematic for routers to maintain this volume of routes in their routing tables. Route summarization, also called route aggregation, is the process of advertising a contiguous set of addresses as a single address with a less specific, shorter subnet mask. This method can reduce the number of routes that a router must maintain because it represents a series of networks as a single summary address.

OSPF route summarization helps solve two major problems: large routing tables and frequent LSA flooding throughout the AS. Every time a route disappears in one area, routers in other areas also get involved in shortest path calculation. To reduce the size of the area database, you can configure summarization on an area boundary or AS boundary.

Normally, type 1 and type 2 LSAs are generated inside each area and translated into type 3 LSAs in other areas. With route summarization, the ABRs or ASBRs consolidate multiple routes into a single advertisement. ABRs summarize type 3 LSAs, and ASBRs summarize type 5 LSAs, as illustrated in Figure 24-4. Instead of advertising many specific prefixes, they advertise only one summary prefix.

Figure 24-4 OSPF Summarization on ABRs and ASBRs

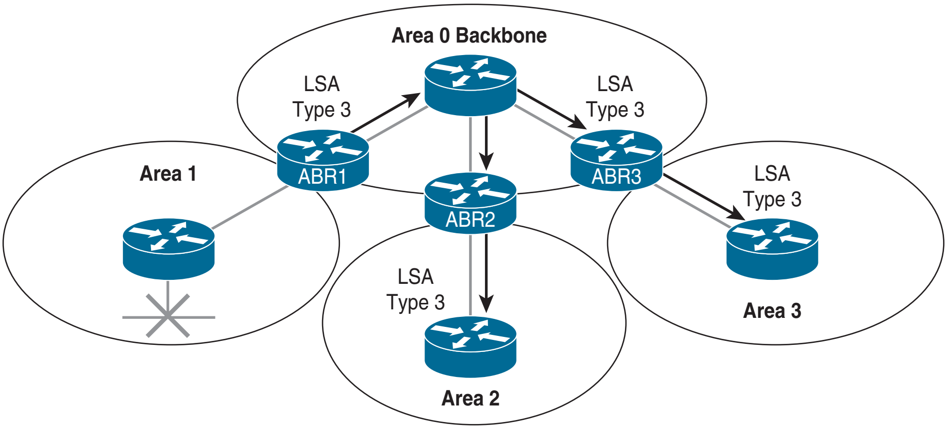

Summarize TYPE 3 LSAs
Summarize TYPE 5 LSAs
Area 0 Backbone
R1
ABR
Area 1
ASBR
External AS EIGRP
To summarize routes at the ABR area boundary, use the area range command.
To summarize routes at the ASBR, use the summary-address command.

If an OSPF design includes multiple ABRs or ASBRs between areas, suboptimal routing is possible. This behavior is one of the drawbacks of summarization.

Route summarization requires a good addressing plan with an assignment of subnets and addresses that lends itself to aggregation at the OSPF area borders. When you summarize routes on a router, it is possible that OSPF still might prefer a different path for a specific network with a longer prefix match than the one proposed by the summary. Also, the summary route has a single metric to represent the collection of routes summarized. This is usually the smallest metric associated with an LSA included in the summary.

Route summarization directly affects the amount of bandwidth, CPU power, and memory resources the OSPF routing process consumes. Route summarization minimizes the number of routing table entries, localizes the impact of a topology change, and reduces LSA flooding and saves CPU resources. Without route summarization, every specific-link LSA is propagated into the OSPF backbone and beyond, causing unnecessary network traffic and router overhead, as illustrated in Figure 24-5, where a LAN interface in Area 1 has failed. This triggers a flooding of type 3 LSAs throughout the OSPF domain.

Figure 24-5 OSPF Type 3 LSA Flooding



With route summarization, only the summarized routes are propagated into the backbone (Area 0). Summarization prevents every router from having to rerun the SPF algorithm, increases the stability of the network, and reduces unnecessary LSA flooding. Also, if a network link fails, the topology change is not propagated into the backbone (and other areas, by way of the backbone). Specific-link LSA flooding outside the area does not occur.

OSPF ABR Route Summarization

With summarization of type 3 summary LSAs, the router creates a summary of all the interarea (type 1 and type 2 LSAs) routes. It is therefore called *interarea route summarization*.

To configure route summarization on an ABR, you use the following command:

```
ABR(config-router)# area area-id range ip-address mask [advertise | not-advertise]
   [cost cost]
```

A summary route is advertised only if you have at least one prefix that falls within the summary range. The ABR that creates the summary route creates a Null0 interface to prevent loops. You can configure a static cost for the summary instead of using the lowest metric from one of the prefixes being summarized. The default behavior is to advertise the summary prefix so the **advertise** keyword is not necessary.

Summarization on an ASBR

It is possible to summarize external networks being advertised by an ASBR. This summarization minimizes the number of routing table entries, reduces type 5 AS external LSA flooding, and saves CPU resources. It also localizes the impact of any topology changes if an external network fails.

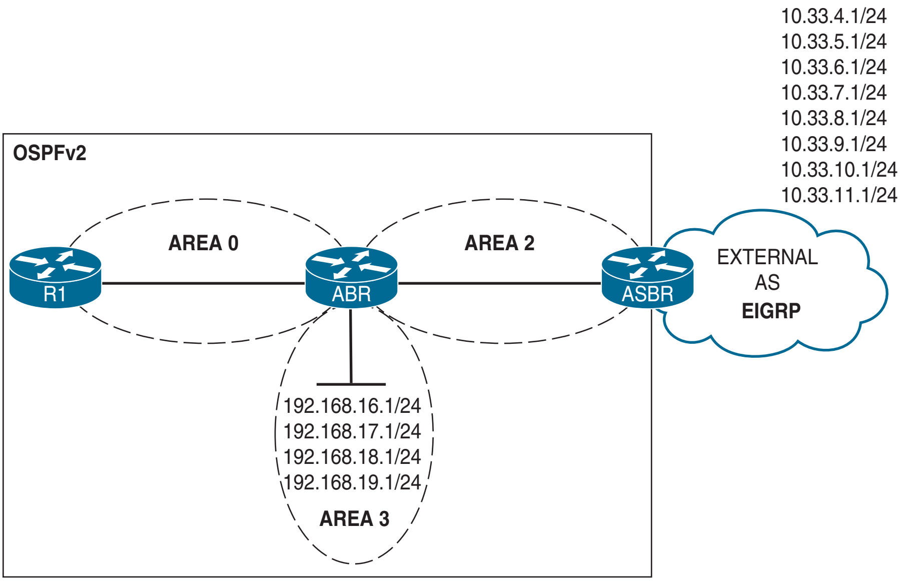
To configure route summarization on an ASBR, use the following command:

```
ASBR(config-router)# summary-address ip-address mask [not-advertise] [tag tag]
  [nssa-only]
```

OSPF Summarization Example

Figure 24-6 shows the topology used in this section's summarization example. The ABR is configured to summarize four prefixes in Area 3, and the ASBR is configured to summarize eight prefixes that originate from the EIGRP external AS.

Figure 24-6 OSPF Summarization Topology Example


10.33.4.1/24
10.33.5.1/24
10.33.6.1/24
10.33.7.1/24
10.33.8.1/24
10.33.9.1/24
10.33.10.1/24
10.33.11.1/24
OSPFv2
AREA 0
AREA 2
R1
ABR
ASBR
EXTERNAL
AS
EIGRP
192.168.16.1/24
192.168.17.1/24
192.168.18.1/24
192.168.19.1/24
AREA 3

Example 24-2 shows the routing table on R1 before summarization. Notice that eight external networks (O E2) and four Area 3 networks (O IA) are present.

Example 24-2 Verifying the Routing Table on R1

```
R1# show ip route ospf
<... output omitted ...>

      10.0.0.0/24 is subnetted, 8 subnets
O E2     10.33.4.0 [110/20] via 172.16.13.2, 01:04:40, GigabitEthernet0/2
```

```
O E2     10.33.5.0 [110/20] via 172.16.13.2, 01:04:40, GigabitEthernet0/2
O E2     10.33.6.0 [110/20] via 172.16.13.2, 01:04:40, GigabitEthernet0/2
O E2     10.33.7.0 [110/20] via 172.16.13.2, 01:04:40, GigabitEthernet0/2
O E2     10.33.8.0 [110/20] via 172.16.13.2, 01:04:40, GigabitEthernet0/2
O E2     10.33.9.0 [110/20] via 172.16.13.2, 01:04:40, GigabitEthernet0/2
O E2     10.33.10.0 [110/20] via 172.16.13.2, 01:04:40, GigabitEthernet0/2
O E2     10.33.11.0 [110/20] via 172.16.13.2, 01:04:40, GigabitEthernet0/2
O IA  192.168.16.0/24 [110/11] via 172.16.13.2, 01:04:40, GigabitEthernet0/2
O IA  192.168.17.0/24 [110/11] via 172.16.13.2, 01:04:40, GigabitEthernet0/2
O IA  192.168.18.0/24 [110/11] via 172.16.13.2, 01:04:40, GigabitEthernet0/2
O IA  192.168.19.0/24 [110/11] via 172.16.13.2, 01:04:40, GigabitEthernet0/2
```

Example 24-3 shows the configuration of summarization on the ABR for the 192.168.16.0/24, 192.168.17.0, 192.168.18.0/24, and 192.168.19.0/24 Area 3 networks into the aggregate route 192.168.16.0/22. Example 24-3 also shows the configuration of summarization on the ASBR for the 10.33.4.0/24 to 10.33.11.0/24 external networks into two aggregate routes, 10.33.4.0/22 and 10.33.8.0/22. Two /22 aggregate routes are used on the ASBR instead of one /21 or one /20 to avoid advertising subnets that don't exist or that don't belong in the external AS.

Example 24-3 Configuring Interarea and External Summarization

```
ABR(config)# router ospf 1
ABR(config-router)# area 3 range 192.168.16.0 255.255.252.0
```

```
ASBR(config)# router ospf 1
ASBR(config-router)# summary-address 10.33.4.0 255.255.252.0
ASBR(config-router)# summary-address 10.33.8.0 255.255.252.0
```

Example 24-4 shows the routing on R1 for verification that the individual longer prefix routes were suppressed and replaced by the interarea route summary (O IA) and the external route summary (O E2).

Example 24-4 Verifying Interarea and External Summarization on R1

```
R1# show ip route ospf

<... output omitted ...>

      10.0.0.0/22 is subnetted, 2 subnets
O E2     10.33.4.0 [110/20] via 172.16.13.2, 00:11:42, GigabitEthernet0/2
O E2     10.33.8.0 [110/20] via 172.16.13.2, 00:11:42, GigabitEthernet0/2
O IA  192.168.16.0/22 [110/11] via 172.16.13.2, 01:00:15, GigabitEthernet0/2
```

OSPF Route Filtering Tools

OSPF has built-in mechanisms for controlling route propagation. OSPF routes are permitted or denied into different OSPF areas based on area type. There are several methods to filter routes on the local router, and the appropriate method depends on whether the router is in the same area or in a different area than the originator of the routes. Most filtering methods do not remove the networks from the LSDB. The routes are removed from the routing table, which prevents the local router from using them to forward traffic. The filters have no impact on the presence of routes in the routing table of any other router in the OSPF routing domain.

Distribute Lists

One of the ways to control routing updates is by using a *distribute list*, which allows you to apply an access list to routing updates. A distribute list filter can be applied to transmitted, received, or redistributed routing updates.

Classic access lists do not affect traffic originated by the router, so applying an access list to an interface has no effect on the outgoing routing advertisements. When you link an access list to a distribute list, routing updates can be controlled no matter what their source.

An access list is configured in global configuration mode and then associated with a distribute list under the routing protocol. An access list should permit the networks that should be advertised or redistributed and deny the networks that should be filtered. The router then applies the access list to the routing updates for that protocol. Options in the **distribute-list** command allow updates to be filtered based on three factors:

- Incoming interface
- Outgoing interface
- Redistribution from another routing protocol

For OSPF, the **distribute-list in** command filters what ends up in the IP routing table—and only on the router on which the **distribute-list in** command is configured. It does not remove routes from the link-state databases of area routers.

It is possible to use a prefix list instead of an access list when matching prefixes for the distribute list. Prefix lists offer better performance than access lists. They can filter based on prefix and prefix length.

Using the **ip prefix-list** command has several benefits in comparison with using the **access-list** command. Prefix lists were intended for use with route filtering, whereas access lists were originally intended to be used for packet filtering.

A router transforms a prefix list into a tree structure, and each branch of the tree serves as a test. Cisco IOS Software determines a verdict of either "permit" or "deny" much faster this way than when sequentially interpreting access lists.

You can assign a sequence number to **ip prefix-list** statements, which gives you the ability to sort statements if necessary. Also, you can add statements at a specific location or delete specific statements. If no sequence number is specified, a default sequence number is applied.

Routers match networks in a routing update against the prefix list by using as many bits as indicated. For example, you can specify a prefix list to be 10.0.0.0/16, which matches 10.0.0.0 routes but not 10.1.0.0 routes.

A prefix list can specify the size of the subnet mask and can also indicate that the subnet mask must be in a specified range.

Prefix lists are similar to access lists in many ways. A prefix list can consist of any number of lines, each of which indicates a test and a result. The router can interpret the lines in the specified order, although Cisco IOS Software optimizes this behavior for processing in a tree structure. When a router evaluates a route against the prefix list, the first line that matches results is either a "permit" or "deny." If none of the lines in the list match, the result is "implicitly deny."

Testing is done using IPv4 or IPv6 prefixes. The router compares the indicated number of bits in the prefix with the same number of bits in the network number in the update. If these numbers match, testing continues, with an examination of the number of bits set in the subnet mask. The **ip prefix-list** command can indicate a prefix length range, and the number must be within that range to pass the test. If you do not indicate a range in the prefix line, the subnet mask must match the prefix size.

OSPF Filtering Options

Internal routing protocol filtering presents some special challenges with link-state routing protocols such as OSPF. Link-state protocols do not advertise routes; instead, they advertise topology information. Also, SPF loop prevention relies on each router in the same area having an identical copy of the LSDB for that area. Filtering or changing LSA contents in transit could conceivably make the LSDBs differ on different routers, causing routing irregularities.

IOS supports four types of OSPF route filtering:

- **ABR type 3 summary LSA filtering using the filter-list command**: This process prevents an ABR from creating certain type 3 summary LSAs.
- **Using the area range not-advertise command**: This process also prevents an ABR from creating specific type 3 summary LSAs.
- **Filtering routes (not LSAs)**: With the **distribute-list in** command, a router can filter the routes that its SPF process is attempting to add to its routing table without affecting the LSDB. This type of filtering can be applied to type 3 summary LSAs and type 5 AS external LSAs.
- **Using the summary-address not-advertise command**: This command is like the **area range not-advertise** command but is applied to the ASBR to prevent it from creating specific type 5 AS external LSAs.

OSPF Filtering: Filter List

ABRs do not forward type 1 and type 2 LSAs from one area into another but instead create type 3 summary LSAs for each subnet defined in the type 1 and type 2 LSAs. Type 3 summary LSAs do not contain detailed information about the topology of the originating area; instead, each type 3 summary LSA represents a subnet and a cost from the ABR to that subnet.

The OSPF ABR type 3 summary LSA filtering feature allows an ABR to filter this type of LSAs at the point where the LSAs would normally be created. By filtering at the ABR, before the type 3 summary LSA is injected into another area, the requirement for identical LSDBs inside the area can be met while still filtering LSAs.

To configure this type of filtering, you use the **area** *area-number* **filter-list prefix** *prefix-list-name* **in | out** command under OSPF configuration mode. The referenced prefix list is used to match the subnets and masks to be filtered. The *area-number* and the **in** | **out** option of the **area filter-list** command work together, as follows:

- When **out** is configured, IOS filters prefixes coming out of the configured area.
- When **in** is configured, IOS filters prefixes going into the configured area.

Returning to the topology illustrated in Figure 24-6, recall that the ABR is currently configured to advertise a summary of Area 3 subnets (192.168.16.0/22). This type 3 summary LSA is flooded into Area 0 and Area 2. In Example 24-5, the ABR router is configured to filter the 192.168.16.0/22 prefix as it enters Area 2. This allows R1 to still receive the summary from Area 3, but the ASBR does not.

Example 24-5 Configuring Type 3 Summary LSA Filtering with a Filter List

```
ABR(config)# ip prefix-list FROM_AREA_3 deny 192.168.16.0/22
ABR(config)# ip prefix-list FROM_AREA_3 permit 0.0.0.0/0 le 32
!
ABR(config)# router ospf 1
ABR(config-router)# area 2 filter-list prefix FROM_AREA_3 in
```

OSPF Filtering: Area Range

The second way to filter OSPF routes is to filter type 3 summary LSAs at an ABR by using the **area range** command. The **area range** command performs route summarization at ABRs, telling a router to cease advertising smaller subnets in a particular address range, instead creating a single type 3 summary LSA whose address and prefix encompass the smaller subnets. When the **area range** command includes the **not-advertise** keyword, not only are the smaller component subnets not advertised as type 3 summary LSAs, but the summary route is also not advertised. As a result, this command has the same effect as the **area filter-list** command with the **out** keyword: It prevents the LSA from going out to any other areas.

Again returning to the topology illustrated in Figure 24-6, instead of using the filter list described previously, Example 24-6 shows the use of the **area range** command to not only filter out the individual Area 3 subnets but also prevent the type 3 summary LSA from being advertised out of Area 3.

Example 24-6 Configuring Type 3 Summary LSA Filtering with Area Range

```
ABR(config)# router ospf 1
ABR(config-router)# area 3 range 192.168.16.0 255.255.252.0 not-advertise
```

The result here is that neither R1 nor the ASBR receives individual Area 3 prefixes or the summary.

OSPF Filtering: Distribute List

For OSPF, the **distribute-list in** command filters what ends up in the IP routing table—and only on the router on which the **distribute-list in** command is configured. It does not remove routes from the link-state database of area routers. The process is straightforward, and the **distribute-list** command can reference either an ACL or a prefix list.

The following rules govern the use of distribute lists for OSPF:

- The distribute list applied in the inbound direction filters results of SPF (the routes to be installed into the router's routing table).
- The distribute list applied in the outbound direction applies only to redistributed routes and only on an ASBR; it selects which redistributed routes to advertise. (Redistribution is beyond the scope of this book.)
- The inbound logic does not filter inbound LSAs; it instead filters the routes that SPF chooses to add to its own local routing table.

In Example 24-7, access list number 10 is used as a distribute list and applied in the inbound direction to filter OSPF routes that are being added to its own routing table.

Example 24-7 Configuring a Distribute List with an Access List

```
R1(config)# access-list 10 deny 192.168.4.0 0.0.0.255
R1(config)# access-list 10 permit any
!
R1(config)# router ospf 1
R1(config-router)# distribute-list 10 in
```

Example 24-8 shows the use of a prefix list with a distribute list to achieve the same result as with commands in Example 24-7.

Example 24-8 Configuring a Distribute List with a Prefix List

```
R1(config)# ip prefix-list seq 5 31DAYS-PFL deny 192.168.4.0/24
R1(config)# ip prefix-list seq 10 31DAYS-PFL permit 0.0.0.0/0 le 32
!
R1(config)# router ospf 1
R1(config-router)# distribute-list prefix 31DAYS-PFL in
```

NOTE: Prefix lists are covered in more detail on Day 23, "BGP."

OSPF Filtering: Summary Address

Recall that type 5 AS external LSAs are originated by an ASBR (router advertising external routes) and flooded through the whole OSPF autonomous system. You cannot limit the way this type of LSA is generated except by controlling the routes advertised into OSPF. When a type 5 AS external LSA is being generated, it uses the RIB contents and honors the **summary-address** commands if configured.

It is then possible to filter type 5 AS external LSAs on the ASBR in much the same way that type 3 summary LSAs are filtered on the ABR. Using the **summary-address not-advertise** command allows you to specify which external networks should be flooded across the OSPF domain as type 5 AS external LSAs.

Returning to the topology illustrated in Figure 24-6, recall that the ASBR router is advertising two type 5 AS external LSAs into the OSPF domain: 10.33.4.0/22 and 10.33.8.0/22. Example 24-9 shows the commands used to prevent the 10.33.8.0/22 type 5 summary or the individual subnets that are part of that summary from being advertised into the OSPF domain.

Example 24-9 Configuring Type 5 AS External LSA Filtering

```
ASBR(config)# router ospf 1
ASBR(config-router)# summary-address 10.33.8.0 255.255.252.0 not-advertise
```

OSPFv3

While OSPFv2 is feature rich and widely deployed, it does have one major limitation in that it does not support the routing of IPv6 networks. Fortunately, OSPFv3 does support IPv6 routing, and it can be configured to also support IPv4 routing.

The traditional OSPFv2 method, which is configured with the **router ospf** command, uses IPv4 as the transport mechanism. The legacy OSPFv3 method, which is configured with the **ipv6 router ospf** command, uses IPv6 as the transport protocol.

The newer OSPFv3 address family framework, which is configured with the **router ospfv3** command, uses IPv6 as the transport mechanism for both IPv4 and IPv6 address families. Therefore, it does not peer with routers running the traditional OSPFv2 protocol. The OSPFv3 address family framework utilizes a single OSPFv3 process. It is capable of supporting IPv4 and IPv6 within that single OSPFv3 process. OSPFv3 builds a single database with LSAs that carry IPv4 and IPv6 information. The OSPF adjacencies are established separately for each address family. Settings that are specific to an address family (IPv4/IPv6) are configured inside that address family router configuration mode.

The OSPFv3 address family framework is supported as of Cisco IOS Release 15.1(3)S and Cisco IOS Release 15.2(1)T. Cisco devices that run software older than these releases and third-party devices do not form neighbor relationships with devices running the address family feature for the IPv4 address family because they do not set the address family bit. Therefore, those devices do not participate in the IPv4 address family SPF calculations and do not install the IPv4 OSPFv3 routes in the IPv6 RIB.

Although OSPFv3 is a rewrite of the OSPF protocol to support IPv6, its foundation remains the same as in IPv4 and OSPFv2. The OSPFv3 metric is still based on interface cost. The packet types and neighbor discovery mechanisms are the same in OSPFv3 as they are for OSPFv2, except for the use of IPv6 link-local addresses. OSPFv3 also supports the same interface types, including broadcast and point-to-point. LSAs are still flooded throughout an OSPF domain, and many of the LSA types are the same, although a few have been renamed or newly created.

More recent Cisco routers support both the legacy OSPFv3 commands (**ipv6 router ospf**) and the newer OSPFv3 address family framework (**router ospfv3**). The focus of this book is on the latter.

Routers that use the legacy OSPFv3 commands should be migrated to the newer commands used in this book. Use the Cisco Feature Navigator (https://cfnng.cisco.com/) to determine compatibility and support.

To start any IPv6 routing protocols, you need to enable IPv6 unicast routing by using the **ipv6 unicast-routing** command.

The OSPF process for IPv6 no longer requires an IPv4 address for the router ID, but it does require a 32-bit number to be set. You define the router ID by using the **router-id** command. If you do not set the router ID, the system tries to dynamically choose an ID from the currently active IPv4 addresses. If there are no active IPv4 addresses, the process fails to start.

In the IPv6 **router ospfv3** configuration mode, you can specify the passive interfaces (using the **passive-interface** command), enable summarization, and fine-tune the operation, but there is no **network** command. Instead, you enable OSPFv3 on interfaces by specifying the address family and the area for that interface to participate in.

The IPv6 address differs from the IPv4 addresses. You have multiple IPv6 interfaces on a single interface: a link-local address and one or more global addresses, among others. OSPF communication within a local segment is based on link-local addresses and not global addresses. These differences are one of the reasons you enable the OSPF process per interface in the interface configuration mode and not with the **network** command.

To enable the OSPF-for-IPv6 process on an interface and assign that interface to an area, use the **ospfv3** *process-id* **[ipv4 | ipv6] area** *area-id* command in interface configuration mode. To be able to enable OSPFv3 on an interface, the interface must be enabled for IPv6. This implementation is typically achieved by configuring a unicast IPv6 address. Alternatively, you could enable IPv6 by using the **ipv6 enable** interface command, which causes the router to derive its link-local address.

By default, OSPF for IPv6 advertises a /128 prefix length for any loopback interfaces that are advertised into the OSPF domain. The **ospfv3 network point-to-point** command ensures that a loopback with a /64 prefix is advertised with the correct prefix length (64 bits) instead of a prefix length of 128.

OSPFv3 LSAs

OSPFv3 renames two LSA types and defines two additional LSA types that do not exist in OSPFv2.

These are the two renamed LSA types:

- **Interarea prefix LSAs for ABRs (type 3):** Type 3 LSAs advertise internal networks to routers in other areas (interarea routes). A type 3 LSA may represent a single network or a set of networks summarized into one advertisement. Only ABRs generate summary LSAs. In OSPFv3, addresses for these LSAs are expressed as prefix/prefix-length instead of address and mask. The default route is expressed as a prefix with length 0.
- **Interarea router LSAs for ASBRs (type 4):** Type 4 LSAs advertise the location of an ASBR. An ABR originates an interarea router LSA into an area to advertise an ASBR that resides outside the area. The ABR originates a separate interarea router LSA for each ASBR it advertises. Routers that are trying to reach an external network use these advertisements to determine the best path to the next hop toward the ASBR.

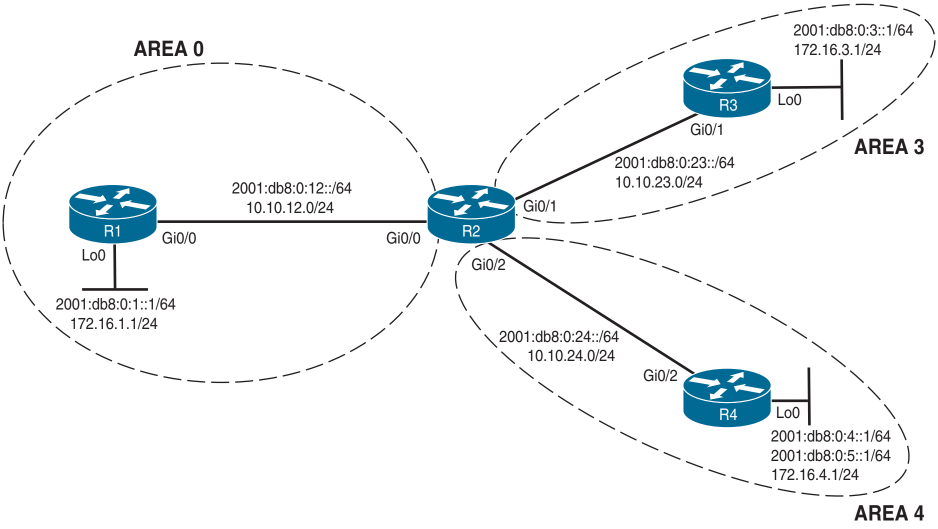
These are the two new LSA types:

- **Link LSAs (type 8):** Type 8 LSAs have local-link flooding scope and are never flooded beyond the link with which they are associated. Link LSAs provide the link-local address of the router to all other routers that are attached to the link. They inform other routers that are attached to the link of a list of IPv6 prefixes to associate with the link. In addition, they allow the router to assert a collection of option bits to associate with the network LSA that will be originated for the link.
- **Intra-area prefix LSAs (type 9):** A router can originate multiple intra-area prefix LSAs for each router or transit network, each with a unique link-state ID. The link-state ID for each intra-area prefix LSA describes its association to either the router LSA or the network LSA. The link-state ID also contains prefixes for stub and transit networks.

OSPFv3 Configuration

Figure 24-7 shows a simple four-router topology to demonstrate multiarea OSPFv3 configuration. An OSPFv3 process can be configured to be IPv4 or IPv6. The **address-family** command is used to determine which AF runs in the OSPFv3 process. Once the address family is selected, you can enable multiple instances on a link and enable address family–specific commands. Loopback 0 is configured as passive under the IPv4 and IPv6 address families. The Loopback 0 interface is also configured with the OSPF point-to-point network type to ensure that OSPF advertises the correct prefix length (/24 for IPv4 and /64 for IPv6). A router ID is also manually configured for the entire OSPFv3 process on each router. R2 is configured to summarize the 2001:db8:0:4::/64 and 2001:db8:0:5::/64 IPv6 prefixes that are configured on R4's Loopback 0 interface. Finally, R2 is configured with a higher OSPF priority to ensure that it is chosen as the DR on all links. Example 24-10 demonstrates the necessary configuration.

Figure 24-7 Multiarea OSPFv3 Configuration


AREA 0
AREA 3
AREA 4
R1
R2
R3
R4
Lo0
2001:db8:0:1::1/64
172.16.1.1/24
Gi0/0
2001:db8:0:12::/64
10.10.12.0/24
Gi0/0
Gi0/1
Gi0/2
Gi0/1
2001:db8:0:23::/64
10.10.23.0/24
Lo0
2001:db8:0:3::1/64
172.16.3.1/24
2001:db8:0:24::/64
10.10.24.0/24
Gi0/2
Lo0
2001:db8:0:4::1/64
2001:db8:0:5::1/64
172.16.4.1/24

Example 24-10 Configuring OSPFv3 for IPv4 and IPv6

```
R1
interface Loopback0
 ip address 172.16.1.1 255.255.255.0
 ipv6 address 2001:DB8:0:1::1/64
 ospfv3 network point-to-point
 ospfv3 1 ipv6 area 0
 ospfv3 1 ipv4 area 0
!
interface Ethernet0/0
 ip address 10.10.12.1 255.255.255.0
 ipv6 address 2001:DB8:0:12::1/64
 ospfv3 1 ipv6 area 0
 ospfv3 1 ipv4 area 0
!
router ospfv3 1
 router-id 1.1.1.1
 !
 address-family ipv4 unicast
  passive-interface Loopback0
 exit-address-family
 !
 address-family ipv6 unicast
  passive-interface Loopback0
 exit-address-family
```

```
R2
interface Ethernet0/0
 ip address 10.10.12.2 255.255.255.0
 ipv6 address 2001:DB8:0:12::2/64
 ospfv3 priority 2
 ospfv3 1 ipv6 area 0
 ospfv3 1 ipv4 area 0
!
interface Ethernet0/1
 ip address 10.10.23.1 255.255.255.0
 ipv6 address 2001:DB8:0:23::1/64
 ospfv3 priority 2
 ospfv3 1 ipv4 area 3
 ospfv3 1 ipv6 area 3
!
interface Ethernet0/2
 ip address 10.10.24.1 255.255.255.0
 ipv6 address 2001:DB8:0:24::1/64
 ospfv3 priority 2
 ospfv3 1 ipv6 area 4
 ospfv3 1 ipv4 area 4
!
router ospfv3 1
 router-id 2.2.2.2
 !
 address-family ipv4 unicast
 exit-address-family
 !
 address-family ipv6 unicast
  area 4 range 2001:DB8:0:4::/63
 exit-address-family
```

R3

```
interface Loopback0
 ip address 172.16.3.1 255.255.255.0
 ipv6 address 2001:DB8:0:3::1/64
 ospfv3 network point-to-point
 ospfv3 1 ipv6 area 3
 ospfv3 1 ipv4 area 3
!
interface Ethernet0/1
 ip address 10.10.23.2 255.255.255.0
 ipv6 address 2001:DB8:0:23::2/64
 ospfv3 1 ipv6 area 3
 ospfv3 1 ipv4 area 3
!
router ospfv3 1
 router-id 3.3.3.3
 !
 address-family ipv4 unicast
  passive-interface Loopback0
 exit-address-family
 !
 address-family ipv6 unicast
  passive-interface Loopback0
 exit-address-family
```

```
R4
interface Loopback0
 ip address 172.16.4.1 255.255.255.0
 ipv6 address 2001:DB8:0:4::1/64
 ipv6 address 2001:DB8:0:5::1/64
 ospfv3 network point-to-point
 ospfv3 1 ipv6 area 4
 ospfv3 1 ipv4 area 4
!
interface Ethernet0/2
 ip address 10.10.24.2 255.255.255.0
 ipv6 address 2001:DB8:0:24::2/64
 ospfv3 1 ipv6 area 4
 ospfv3 1 ipv4 area 4
!
router ospfv3 1
 router-id 4.4.4.4
 !
 address-family ipv4 unicast
  passive-interface Loopback0
 exit-address-family
 !
 address-family ipv6 unicast
  passive-interface Loopback0
 exit-address-family
```

In Example 24-10, observe the following highlighted configuration commands:

- The **ospfv3 network point-to-point** command is applied to the Loopback 0 interface on R1, R3, and R4.
- Each router is configured with a router ID under the global OSPFv3 process using the **router-id** command.
- The **passive-interface** command is applied under each OSPFv3 address family on R1, R3, and R4 for Loopback 0.
- The **ospfv3 priority 2** command is entered on R2's Ethernet interfaces to ensure that it is chosen as the DR. R1, R3, and R4 then become BDRs on the link they share with R2.
- The **area range** command is applied to the OSPFv4 IPv6 address family on R2 because it is the ABR in the topology. The command summarizes the Area 4 Loopback 0 IPv6 addresses on R4. The result is that a type 3 interarea prefix LSA is advertised into Area 0 and Area 3 for the 2001:db8:0:4/63 prefix.
- Individual router interfaces are placed in the appropriate area for the IPv4 and IPv6 address families using the **ospfv3 ipv4 area** and **ospfv3 ipv6 area** commands. OSPFv3 is configured to use process ID 1.

OSPFv3 Verification

Example 24-11 shows the following verification commands: **show ospfv3 neighbor**, **show ospfv3 interface brief**, **show ip route ospfv3**, and **show ipv6 route ospf**. Notice that the syntax for each of the OSPFv3 verification commands is practically identical to that of its OSPFv2 counterpart.

Example 24-11 Verifying OSPFv3 for IPv4 and IPv6

```
R2# show ospfv3 neighbor

          OSPFv3 1 address-family ipv4 (router-id 2.2.2.2)

Neighbor ID     Pri   State         Dead Time    Interface ID     Interface
1.1.1.1           1   FULL/BDR      00:00:31     3                GigabitEthernet0/0
3.3.3.3           1   FULL/BDR      00:00:34     4                GigabitEthernet0/1
4.4.4.4           1   FULL/BDR      00:00:32     5                GigabitEthernet0/2

          OSPFv3 1 address-family ipv6 (router-id 2.2.2.2)

Neighbor ID     Pri   State         Dead Time    Interface ID     Interface
1.1.1.1           1   FULL/BDR      00:00:33     3                GigabitEthernet0/0
3.3.3.3           1   FULL/BDR      00:00:31     4                GigabitEthernet0/1
4.4.4.4           1   FULL/BDR      00:00:34     5                GigabitEthernet0/2


R2# show ospfv3 interface brief
Interface      PID   Area             AF          Cost  State Nbrs F/C
Gi0/0          1     0                ipv4        1     DR    1/1
Gi0/1          1     3                ipv4        1     DR    1/1
Gi0/2          1     4                ipv4        1     DR    1/1
Gi0/0          1     0                ipv6        1     DR    1/1
Gi0/1          1     3                ipv6        1     DR    1/1
Gi0/2          1     4                ipv6        1     DR    1/1

R1# show ip route ospfv3
<. . . output omitted . . .>
Gateway of last resort is not set

      10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
O IA     10.10.23.0/24 [110/2] via 10.10.12.2, 00:13:47, GigabitEthernet0/0
O IA     10.10.24.0/24 [110/2] via 10.10.12.2, 00:13:47, GigabitEthernet0/0
      172.16.0.0/16 is variably subnetted, 4 subnets, 2 masks
O IA     172.16.3.0/24 [110/3] via 10.10.12.2, 00:13:47, GigabitEthernet0/0
O IA     172.16.4.0/24 [110/3] via 10.10.12.2, 00:13:47, GigabitEthernet0/0
```

```
R1# show ipv6 route ospf
IPv6 Routing Table - default - 9 entries
<. . . output omitted . . .>

OI  2001:DB8:0:3::/64 [110/3]
     via FE80::A8BB:CCFF:FE00:200, GigabitEthernet0/0
OI  2001:DB8:0:4::/63 [110/3]
     via FE80::A8BB:CCFF:FE00:200, GigabitEthernet0/0
OI  2001:DB8:0:23::/64 [110/2]
     via FE80::A8BB:CCFF:FE00:200, GigabitEthernet0/0
OI  2001:DB8:0:24::/64 [110/2]
     via FE80::A8BB:CCFF:FE00:200, GigabitEthernet0/0
```

In Example 24-11, the **show ospfv3 neighbor** and **show ospfv3 interface brief** commands are executed on R2, which is the ABR. Notice that these commands provide output for both the IPv4 and IPv6 address families. The output confirms the DR and BDR status of each OSPF router.

The **show ip route ospfv3** and **show ipv6 route ospf** commands are executed on R1. Notice the cost of 3 for R1 to reach the loopback interfaces on R3 and R5. The total cost is calculated as follows: The link from R1 to R2 has a cost of 1, the link from R2 to either R3 or R4 has a cost of 1, and the default cost of a loopback interface in OSPFv2 or OSPFv3 is 1, for a total of 3. All OSPF entries on R1 are considered O IA because they are advertised to R1 by R2 using a type 3 interarea prefix LSA. The 2001:db8:0:4::/63 prefix is the summary configured on R2.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	8, 9, 10
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	5

Day 23

BGP

ENCOR 350-401 Exam Topics

Infrastructure

- Layer 3
  - Configure and verify eBGP between directly connected neighbors (best path selection algorithm and neighbor relationships)

Key Topics

Today we review Border Gateway Protocol (BGP), which is used as the routing protocol to exchange routes between autonomous systems (ASs). BGP, which is defined in RFC 4271, is a routing protocol that is widely used in MPLS implementations and is the underlying routing foundation of the Internet. This protocol is complex and scalable, and it is also reliable and secure. We will explore the concept of interdomain routing with BGP and configuration of a single-homed External Border Gateway Protocol (EBGP) connection between a customer and a service provider.

BGP Interdomain Routing

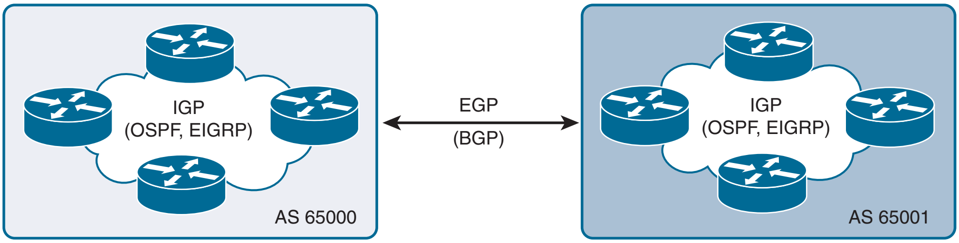
BGP is a routing protocol used to exchange information between autonomous systems (ASs). An AS is defined as a collection of networks under a single technical administration domain. Other definitions refer to an AS as a collection of routers or IP prefixes, but in the end, the definitions are all essentially the same. The important principle is the technical administration, which means routers that share the same routing policy. Legal and administrative ownership of the routers does not matter with autonomous systems.

Autonomous systems are identified by AS numbers. An AS number is a 16-bit integer ranging from 1 to 65,535. Public AS numbers (1 to 64,511) are assigned and managed by the Internet Assigned Numbers Authority (IANA). A range of private AS numbers (64,512 to 65,535) has been reserved for customers that need AS numbers to run BGP in their private networks. New 32-bit AS numbers were created when the AS number pool approached exhaustion.

To understand BGP, you must first understand how it differs from other routing protocols. One way to categorize routing protocols is based on whether they are interior or exterior, as illustrated in Figure 23-1:

- An Interior Gateway Protocol (IGP) is a routing protocol that exchanges routing information within an AS. Routing Information Protocol (RIP), Open Shortest Path First (OSPF), Enhanced Interior Gateway Routing Protocol (EIGRP), and Intermediate System-to-Intermediate System (IS-IS) are examples of IGPs.
- An Exterior Gateway Protocol (EGP) is a routing protocol that exchanges routing information between different autonomous systems. BGP is an example of an EGP.

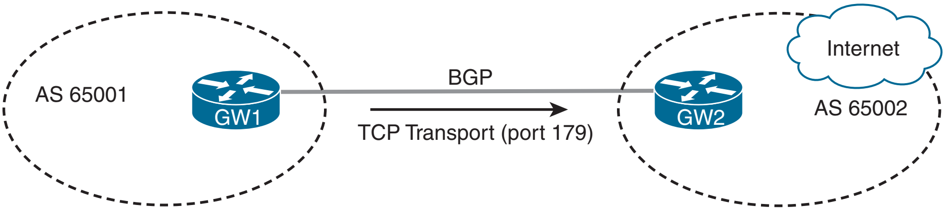
Figure 23-1 IGP vs. EGP



BGP Characteristics

BGP uses TCP as the transport mechanism on port 179, as illustrated in Figure 23-2, which provides reliable connection-oriented delivery. Therefore, BGP does not have to implement retransmission or error recovery mechanisms.

Figure 23-2 BGP and TCP



After the connection is made, BGP peers exchange complete routing tables. However, because the connection is reliable, BGP peers send only changes (incremental, or triggered, updates) after the initial connection. Reliable links do not require periodic routing updates, so routers use triggered updates instead.

BGP sends keepalive messages, similar to the hello messages that OSPF and EIGRP send. IGPs have their own internal function to ensure that the update packets are explicitly acknowledged. These protocols use a one-for-one window, so that if either OSPF or EIGRP has multiple packets to send, the next packet cannot be sent until OSPF or EIGRP receives an acknowledgment from the first update packet. This process can be inefficient and can cause latency issues if thousands of update packets must be exchanged over relatively slow serial links. OSPF and EIGRP rarely have thousands of update packets to send.

BGP is capable of handling the entire Internet table of more than 800,000 networks, and it uses TCP to manage the acknowledgment function. TCP uses a dynamic window, which allows 65,576 bytes to be outstanding before it stops and waits for an acknowledgment. For example, if 1000-byte packets are being sent, there would need to be 65 packets that have not been acknowledged for BGP to stop and wait for an acknowledgment when using the maximum window size. TCP is designed to use a sliding window. The receiver acknowledges the received packets at the halfway point of the sending window. This method allows any TCP application, such as BGP, to continue to stream packets without having to stop and wait, as would be required with OSPF or EIGRP.

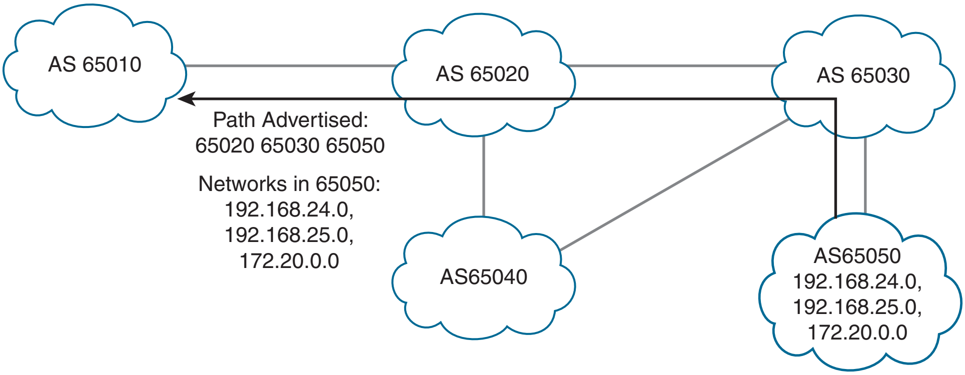
Unlike OSPF and EIGRP, which send changes in topology immediately when they occur, BGP sends batched updates so that the flapping of routes in one autonomous system does not affect all the others. The trade-off is that BGP is relatively slow to converge compared to IGPs such as EIGRP and OSPF. BGP also offers mechanisms that suppress the propagation of route changes if the networks' availability status changes too often.

BGP Path Vector Functionality

BGP routers exchange network layer reachability information (NLRI), called path vectors, which are made up of prefixes and their path attributes.

The path vector information includes a list of the complete hop-by-hop path of BGP AS numbers that are necessary to reach a destination network and the networks that are reachable at the end of the path, as illustrated in Figure 23-3. Other attributes include the IP address to get to the next AS (the next-hop attribute) and an indication of how the networks at the end of the path were introduced to BGP (the origin code attribute).

Figure 23-3 BGP Path Vector



This AS path information is useful for constructing a graph of loop-free autonomous systems and is used to identify routing policies so that restrictions on routing behavior can be enforced, based on the AS path.

The AS path is always loop free. A router that is running BGP does not accept a routing update that already includes its AS number in the path list because the update has already passed through its AS, and accepting it again would result in a routing loop.

An administrator can define policies or rules about how data will flow through the autonomous systems.

BGP Routing Policies

BGP allows you to define routing policy decisions at the AS level. These policies can be implemented for all networks that are owned by an AS, for a certain classless interdomain routing (CIDR) block of network numbers (prefixes), or for individual networks or subnetworks.

BGP specifies that a router can advertise to neighboring autonomous systems only routes that it uses itself. This rule reflects the hop-by-hop routing paradigm that the Internet generally uses. This routing paradigm does not support all possible policies. For example, BGP does not enable one AS to send traffic to a neighboring AS, intending that the traffic takes a different route from the path that is taken by traffic that originates in that neighboring AS. In other words, how a neighboring AS routes traffic cannot be influenced, but how traffic gets to a neighboring AS can be influenced. However, BGP supports any policy that conforms to the hop-by-hop routing paradigm.

Because the Internet uses the hop-by-hop routing paradigm, and because BGP can support any policy that conforms to this model, BGP is highly applicable as an inter-AS routing protocol.

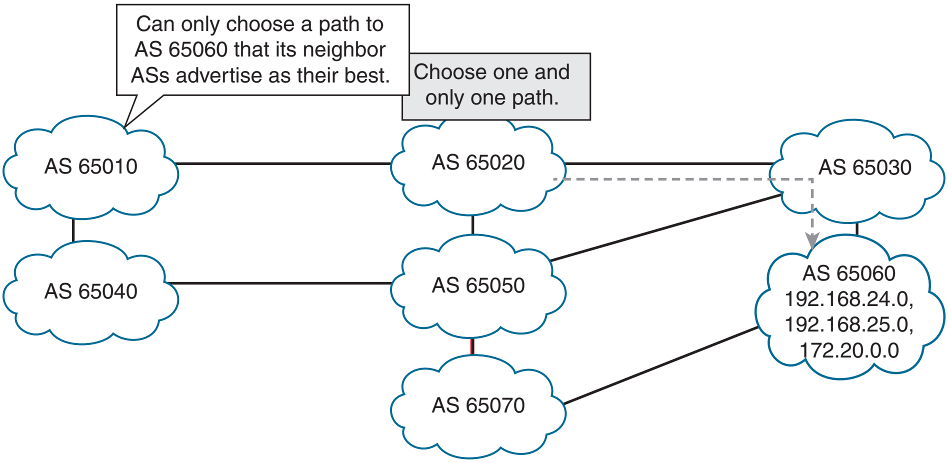
Design goals for interdomain routing with BGP include the following:

- Scalability:
    - BGP exchanges more than 800,000 aggregated Internet routes, and the number of routes is still growing.
- Secure routing information exchange:
    - Routers from another AS cannot be trusted, so BGP neighbor authentication is desirable.
    - Tight route filters are required. For example, it is important with BGP that a multihomed customer AS does not become a transit AS for its providers.
- Support for routing policies:
    - Routing between autonomous systems might not always follow the optimum path. BGP routing policies must address both outgoing and incoming traffic flows.
    - Exterior routing protocols such as BGP have to support a wide range of customer routing requirements.

In Figure 23-4, the following paths are possible for AS 65010 to reach networks in AS 65060 through AS 65020:

- 65020–65030–65060
- 65020–65050–65030–65060
- 65020–65050–65070–65060
- 65020–65030–65050–65070–65060

Figure 23-4 BGP Hop-by-Hop Path Selection


Can only choose a path to AS 65060 that its neighbor ASs advertise as their best.
Choose one and only one path.
AS 65010
AS 65020
AS 65030
AS 65040
AS 65050
AS 65060
192.168.24.0,
192.168.25.0,
172.20.0.0
AS 65070

AS 65010 does not see all these possibilities.

AS 65020 advertises to AS 65010 only its best path, 65020–65030–65060, the same way that IGPs announce only their best least-cost routes. For BGP, a shorter AS path is preferred over a longer AS path. The path 65020–65030–65060 is the only path through AS 65020 that AS 65010 sees. All packets that are destined for 65060 through 65020 take this path.

Even though other paths exist, AS 65010 can only use what AS 65020 advertises for the networks in AS 65060. The AS path that is advertised, 65020–65030–65060, is the AS-by-AS (hop-by-hop) path that AS 65020 uses to reach the networks in AS 65060. AS 65020 does not announce another path, such as 65020–65050–65030–65060, because it did not choose that as the best path, based on the BGP routing policy in AS 65020.

AS 65010 does not learn about the second-best path or any other paths from AS 65020 unless the best path of AS 65020 becomes unavailable. Even if AS 65010 were aware of another path through AS 65020 and wanted to use it, AS 65020 would not route packets along that other path because AS 65020 selected 65030–65060 as its best path, and all AS 65020 routers use that path as a matter of BGP policy. BGP does not let one AS send traffic to a neighboring AS, intending for the traffic to take a different route from the path that is taken by traffic originating in the neighboring AS.

To reach the networks in AS 65060, AS 65010 can choose to use AS 65020, or it can choose to go through the path that AS 65040 is advertising. AS 65010 selects the best path to take based on it is own BGP routing policies. The path through AS 65040 is still longer than the path through AS 65020, so AS 65010 prefers the path through AS 65020 unless a different routing policy is put in place in AS 65010.

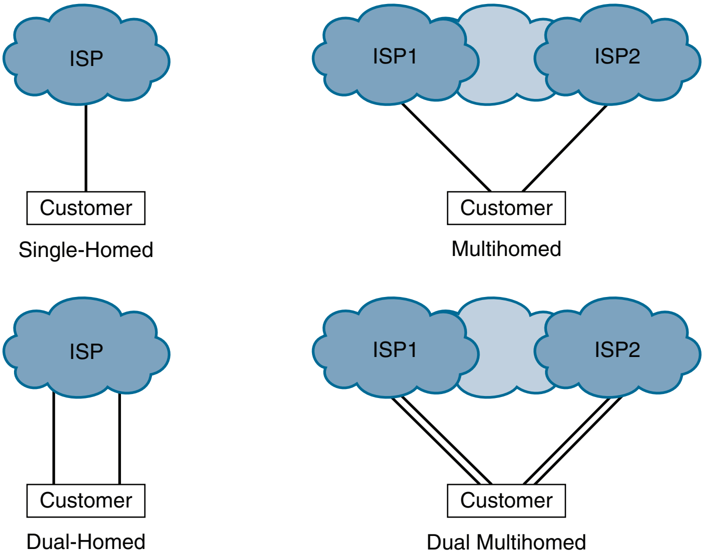
BGP Multihoming

There are multiple strategies for connecting a corporate network to an ISP. The topology depends on the needs of the company.

There are various names for the different types of connections, as illustrated in Figure 23-5:

- **Single-homed:** With a connection to a single ISP when no link redundancy is used, the customer is single-homed. If the ISP network fails, connectivity to the Internet is interrupted. This option is rarely used for corporate networks.
- **Dual-homed:** With a connection to a single ISP, redundancy can be achieved if two links toward the same ISP are used effectively. This is called being dual-homed. There are two options for dual homing: Both links can be connected to one customer router or, to enhance the resiliency further, the two links can terminate at separate routers in the customer's network. In either case, routing must be properly configured to allow both links to be used.
- **Multihomed:** With connections to multiple ISPs, redundancy is built into the design. A customer connected to multiple ISPs is said to be multihomed and is thus resistant to a single ISP failure. Connections from different ISPs can terminate on the same router or on different routers to further enhance the resiliency. The customer is responsible for announcing its own IP address space to upstream ISPs, but it should avoid forwarding any routing information between ISPs (to prevent the customer from becoming a transit provider between the two ISPs). The routing used must be capable of reacting to dynamic changes. Multihoming makes possible load balancing of traffic between ISPs.
- **Dual multihomed:** To further enhance the resiliency with connections to multiple ISPs, a customer can have two links toward each ISP. This solution is called being dual multihomed and typically involves multiple edge routers, one per ISP. As with the dual-homed option, the dual multihomed option can support two links to two different customer routers.

Figure 23-5 BGP Multihoming Options



BGP Operations

Similar to other IGP protocols, BGP maintains relevant neighbor and route information and exchanges different types of messages to create and maintain an operational routing environment.

BGP Data Structures

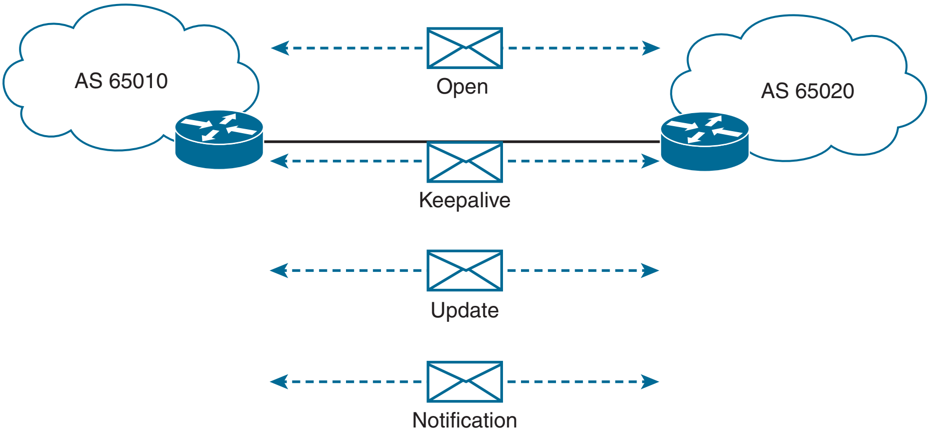
A router that is running BGP keeps its own tables to store BGP information that it receives from and sends to other routers, including a neighbor table and a BGP table (also called a forwarding database or topology database). BGP also uses the IP routing table to forward the traffic. These three tables are described as follows:

- **BGP neighbor table:** For BGP to establish an adjacency, it must be explicitly configured with each neighbor. BGP forms a TCP relationship with each of the configured neighbors and keeps track of the state of these relationships by periodically sending BGP/TCP keepalive messages.
- **BGP table:** After establishing an adjacency, the neighbors exchange the BGP routes. Each router collects these routes from each neighbor that successfully establishes an adjacency and then places the routes in its BGP forwarding database. The best route for each network is selected from the BGP forwarding database, using the BGP route selection process, and is then offered to the IP routing table.
- **IP routing table:** Each router compares the offered BGP routes to any other possible paths to those networks. Then, the best route, based on administrative distance, is installed in the IP routing table. External BGP routes (BGP routes that are learned from an external AS) have an administrative distance of 20. Internal BGP routes (BGP routes that are learned from within the AS) have an administrative distance of 200.

BGP Message Types

As illustrated in Figure 23-6, there are four types of BGP messages: OPEN, KEEPALIVE, UPDATE, and NOTIFICATION.

Figure 23-6 BGP Message Types



After a TCP connection is established, the first message that is sent by each side is an OPEN message. If the OPEN message is acceptable, the side that receives the message sends a KEEPALIVE confirmation. After the receiving side confirms the OPEN message and establishes the BGP connection, the BGP peers can exchange any UPDATE, KEEPALIVE, and NOTIFICATION messages.

An OPEN message includes the following information:

- **Version number:** The suggested version number. The highest common version that both routers support is used. Most BGP implementations today use BGP4.
- **AS number:** The AS number of the local router. The peer router verifies this information. If it is not the AS number that is expected, the BGP session is ended.
- **Hold time:** The maximum number of seconds that can elapse between the successive KEEPALIVE and UPDATE messages from the sender. On receipt of an OPEN message, the router calculates the value of the hold timer by using whichever is smaller: its own configured hold time or the hold time that was received in the OPEN message from the neighbor.
- **BGP router ID:** A 32-bit field that indicates the BGP ID of the sender. The BGP ID is an IP address that is assigned to the router, and it is determined at startup. The BGP router ID is chosen in the same way that the OSPF router ID is chosen: It is the highest active IP address on the router unless a loopback interface with an IP address exists. In that case, the router ID is the highest loopback IP address. The router ID can also be manually configured.
- **Optional parameters:** Parameters that are Type Length Value (TLV) encoded. An example of an optional parameter is session authentication.

BGP peers send KEEPALIVE messages to ensure that connections between the BGP peers still exist. KEEPALIVE messages are exchanged between BGP peers frequently enough to keep the hold timer from expiring. If the negotiated hold time interval is 0, then periodic KEEPALIVE messages are not sent. A KEEPALIVE message consists of only a message header.

BGP peers initially exchange their full BGP routing tables by using an UPDATE message. Incremental updates are sent only after topology changes in the network occur. A BGP UPDATE message has information that is related to one path only; multiple paths require multiple UPDATE messages. All the attributes in the UPDATE message refer to that path and the networks that can be reached through that path.

An UPDATE message can include the following fields:

- **Withdrawn routes:** This list displays IP address prefixes for routes that are withdrawn from service, if any.
- **Path attributes:** These attributes include the AS path, origin, local preference, and so on. Each path attribute includes the attribute TLV. The attribute type consists of the attribute flags, followed by the attribute type code.
- **Network layer reachability information (NLRI):** This field contains a list of IP address prefixes that are reachable by this path.

A BGP NOTIFICATION message is sent when an error condition is detected. The BGP connection is closed immediately after this NOTIFICATION message is sent. A NOTIFICATION message includes an error code, an error subcode, and data that is related to the error.

BGP Neighbor States

Table 23-1 lists the various BGP states. If all works well, a neighbor relationship reaches the final state: Established. When the neighbor relationship (also called a BGP peer or BGP peer connection) reaches the Established state, the neighbors can send BGP UPDATE messages, which list path ttributes and prefixes. However, if the neighbor relationship fails for some reason, the neighbor relationship can cycle through all the states listed in Table 23-1 while the routers periodically attempt to bring up the peering session.

Table 23-1 BGP Neighbor States

State	Typical Reason
Idle	The BGP process is either administratively down or awaiting the next retry attempt.
Connect	The BGP process is waiting for the TCP connection to be completed. You cannot determine from this state whether the TCP connection can complete.
Active	The TCP connection has been completed, but no BGP messages have yet been sent to the peer.
OpenSent	The TCP connection exists, and a BGP OPEN message has been sent to the peer, but the matching OPEN message has not yet been received from the other router.
OpenConfirm	An OPEN message has been both sent to and received from the other router. The next step is to receive a BGP KEEPALIVE message (to confirm that all the neighbor-related parameters match) or a BGP NOTIFICATION message (to learn that there is some mismatch in neighbor parameters).
Established	All neighbor parameters match, the neighbor relationship works, and the peers can now exchange UPDATE messages.

If the router is in the active state, it has found the IP address in the neighbor statement and has created and sent out a BGP open packet. However, the router has not received a response (open confirm packet). One common problem in this case is that the neighbor may not have a return route to the source IP address.

Another common problem that is associated with the active state occurs when a BGP router attempts to peer with another BGP router that does not have a **neighbor** statement peering back to the first router or when the other router is peering with the wrong IP address on the first router. Check to ensure that the other router has a **neighbor** statement that is peering to the correct address of the router that is in the active state.

If the state toggles between the idle state and the active state, one of the most common problems is AS number misconfiguration.

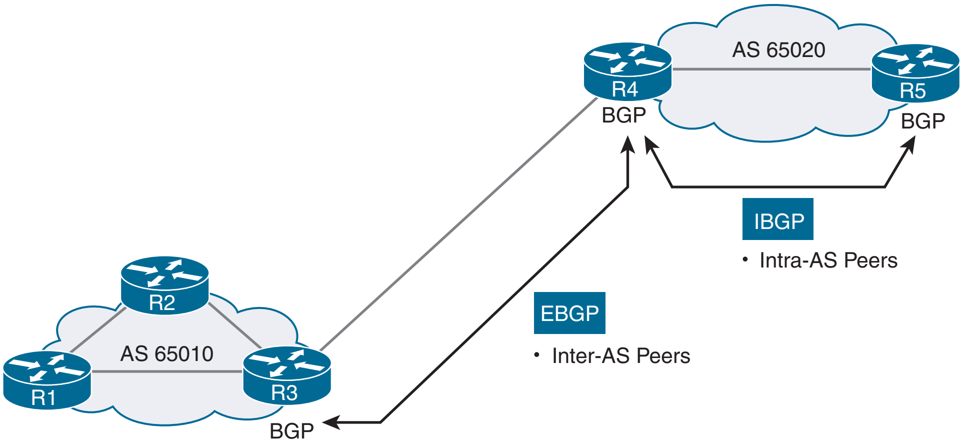
BGP Neighbor Relationships

A BGP router forms a neighbor relationship with a limited number of other BGP routers. Through these BGP neighbors, a BGP router learns paths to reach any advertised enterprise or Internet network.

Any router that runs BGP is known as a *BGP speaker*. The term *BGP peer* has a specific meaning: It is a BGP speaker that is configured to form a neighbor relationship with another BGP speaker so they can directly exchange BGP routing information with each other. A BGP speaker has a limited number of BGP neighbors with which it peers and forms TCP-based relationships.

BGP peers are also known as *BGP neighbors* and can be either internal or external to the AS, as illustrated in Figure 23-7.

Figure 23-7 BGP Neighbor Types



When BGP is running within the same autonomous system, it is called Internal Border Gateway Protocol (IBGP). IBGP is widely used within providers' autonomous systems for redundancy and load-balancing purposes. IBGP peers can be either directly or indirectly connected.

When BGP is running between routers in different autonomous systems, as it is in interdomain routing, it is called External Border Gateway Protocol (EBGP).

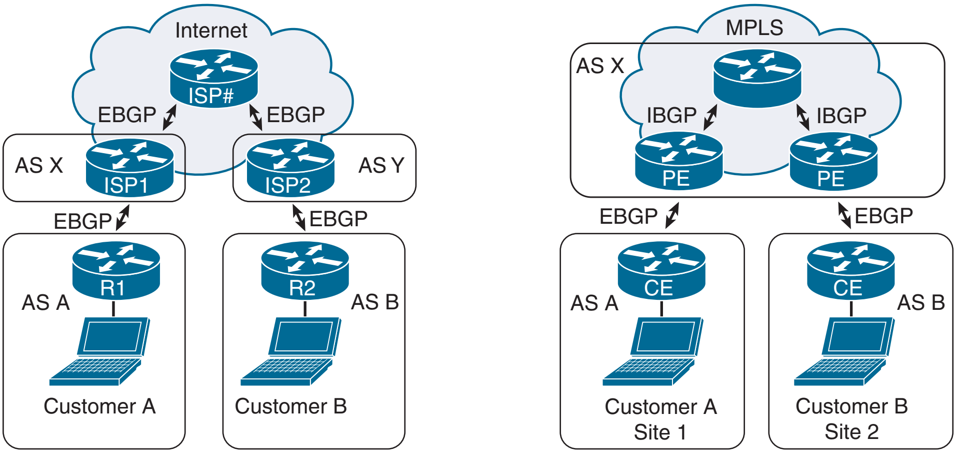
NOTE: According to RFC 4271, the preferred acronyms are IBGP and EBGP, instead of iBGP and eBGP.

EBGP and IBGP

An EBGP peer forms a neighbor relationship with a router in a different AS. Customers use EBGP to exchange routes between their local autonomous systems and their providers.

With Internet connectivity, EBGP is used to advertise internal customer routes to the Internet through multiple ISPs. ISPs use EBGP to exchange routes with other ISPs as well, as illustrated in Figure 23-8.

Figure 23-8 EBGP Neighbors



EBGP is also commonly run between customer edge (CE) and provider edge (PE) routers to exchange enterprise routes between customer sites through a Multiprotocol Label Switching (MPLS) cloud. IBGP can be used inside the MPLS provider cloud to carry customer routes between sites.

Requirements for establishing an EBGP neighbor relationship include the following:

- **Different AS number:** EBGP neighbors must reside in different autonomous systems to be able to form an EBGP relationship.
- **Defined neighbors:** A TCP session must be established before BGP routing update exchanges begin.
- **Reachability:** By default, EBGP neighbors must be directly connected, and the IP addresses on the link must be reachable from each AS.

The requirements for IBGP are identical to those for EBGP except that IBGP neighbors must reside in the same AS to be able to form an IBGP relationship.

BGP Path Selection

Companies that offer mission-critical business services often like to have their networks redundantly connected using either multiple links to the same ISP or using links to different ISPs. Companies that calculate the expected loss of business in the event of an unexpected disconnection may conclude that having two connections is profitable. In such cases, the company may consider being a customer of two different providers or having two separate connections to one provider.

In a multihomed deployment, BGP routers have several peers and receive routing updates from each neighbor. All routing updates enter the BGP forwarding table, and as a result, multiple paths may exist to reach a given network.

Paths for the network are evaluated to determine the best path. Paths that are not the best are eliminated from the selection criteria but are kept in the BGP forwarding table in case the best path becomes inaccessible. If one of the best paths is not accessible, a new best path must be selected.

BGP is not designed to perform load balancing: Paths are chosen based on the policy and not based on link characteristics such as bandwidth, delay, or utilization. The BGP selection process eliminates any multiple paths until a single best path remains.

The BGP best path is evaluated against any other routing protocols that can also reach that network. The route from the source with the lowest administrative distance is installed in the routing table.

BGP Route Selection Process

After BGP receives updates about different destinations from different autonomous systems, it chooses the single best path to reach a specific destination.

Routing policy is based on factors called *attributes*. The following process summarizes how BGP chooses the best route on a Cisco router:

1. Prefer highest weight attribute (local to router).
2. Prefer highest local preference attribute (global within AS).
3. Prefer route originated by the local router (next hop = 0.0.0.0).
4. Prefer shortest AS path (least number of autonomous systems in AS_Path attribute).
5. Prefer lowest origin attribute (IGP < EGP < incomplete).
6. Prefer lowest MED attribute (exchanged between autonomous systems).
7. Prefer an EBGP path over an IBGP path.
8. (IBGP route) Prefer path through the closest IGP neighbor (best IGP metric.)
9. (EBGP route) Prefer oldest EBGP path (neighbor with longest uptime.)
10. Prefer path with the lowest neighbor BGP router ID.
11. Prefer path with the lowest neighbor IP address (multiple paths to same neighbor).

When faced with multiple routes to the same destination, BGP chooses the best route for routing traffic toward the destination by following this route selection process. For example, suppose that there are seven paths to reach network 10.0.0.0. No paths have AS loops, and all paths have valid next-hop addresses, so all seven paths proceed to step 1, which examines the weight of the paths. All seven paths have a weight of 0, so all paths proceed to step 2, which examines the local preference of the paths. Four of the paths have a local preference of 200, and the other three have local preferences of 100, 100, and 150. The four with a local preference of 200 continue the evaluation process in the next step. The other three are still in the BGP forwarding table but are disqualified from being the best path in this case. BGP continues the evaluation process until only a single best path remains. The single best path that remains is submitted to the IP routing table as the best BGP path.

BGP Path Attributes

Routes that are learned via BGP have specific properties known as *BGP path attributes*. These attributes help with calculating the best route when multiple paths to a particular destination exist.

There are two major types of BGP path attributes:

- Well-known BGP attributes
- Optional BGP attributes

Well-Known BGP Attributes

Well-known attributes are attributes that all BGP routers are required to recognize and to use in the path determination process.

There are two categories of well-known attributes: mandatory and discretionary.

Well-Known Mandatory Attributes

Well-known mandatory attributes, which are required to be present for every route in every update, include the following:

- **Origin:** When a router first originates a route in BGP, it sets the origin attribute. If information about an IP subnet is injected using the **network** command or via aggregation (route summarization in BGP), the origin attribute is set to I for IGP. If information about an IP subnet is injected using redistribution, the origin attribute is set to ? for unknown or incomplete information (these two terms have the same meaning). The origin code e was used when the Internet was migrating from EGP to BGP and is now obsolete.
- **AS_Path:** This attribute is a sequence of AS numbers through which the network is accessible.
- **Next_Hop:** This attribute indicates the IP address of the next-hop router. The next-hop router is the router to which the receiving router should forward the IP packets to reach the destination that is advertised in the routing update. Each router modifies the next-hop attribute as the route passes through the network.

Well-Known Discretionary Attributes

Well-known discretionary attributes may or may not be present for a route in an update. Routers use well-known discretionary attributes only when certain functions are required to support the desired routing policy. Examples of well-known discretionary attributes include the following:

- **Local preference:** Local preference is used to achieve a consistent routing policy for traffic exiting an AS.
- **Atomic aggregate:** The atomic aggregate attribute is attached to a route that is created as a result of route summarization (called *aggregation* in BGP). This attribute signals that information that was present in the original routing updates may have been lost when the updates were summarized into a single entry.

Optional BGP Attributes

Optional attributes are attributes that BGP implementations do not require in order for the router to determine the best path. These attributes are either specified in a later extension of BGP or in private vendor extensions that are not documented in a standards document.

When a router receives an update that contains an optional attribute, the router checks to see whether its implementation recognizes the particular attribute. If it does, the router should know how to use it to determine the best path and whether to propagate it. If the router does not recognize an optional attribute, it looks at the transitive bit to determine what category of optional attribute it is.

There are two categories of optional attributes: transitive and nontransitive.

Optional Transitive Attributes

Optional transitive attributes, although not recognized by a router, might still be helpful to upstream routers. These attributes are propagated even when they are not recognized. If a router propagates an unknown transitive optional attribute, it sets an extra bit in the attribute header. This bit is called the partial bit, and it indicates that at least one of the routers in the path did not recognize the meaning of a transitive optional attribute. Examples of optional transitive attributes include the following:

- **Aggregator:** This attribute identifies the AS and the router within the AS that created a route summarization, or aggregate.
- **Community:** This attribute is a numeric value that can be attached to certain routes when they pass a specific point in the network. For filtering or route selection purposes, other routers can examine the community value at different points in the network. BGP configuration may cause routes with a specific community value to be treated differently than others.

Optional Nontransitive Attributes

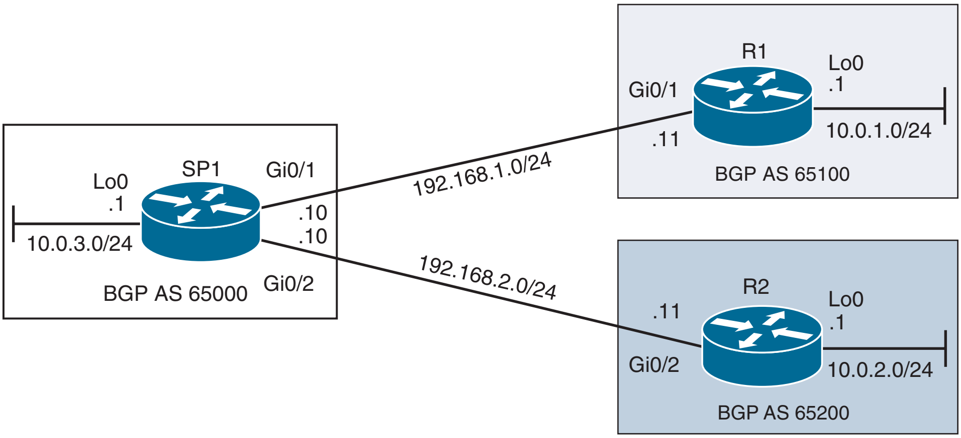
If a router receives a route with an optional nontransitive attribute but does not know how to use it to determine the best path, it drops the attribute before advertising the route. The following is an example of an optional nontransitive attribute:

- **MED:** This attribute influences inbound traffic to an AS from another AS with multiple entry points.

BGP Configuration

Figure 23-9 shows the topology for the BGP configuration example that follows. The focus in this example is a simple EBGP scenario with a service provider router (SP1) and two customer routers (R1 and R2). Separate EBGP sessions are established between the SP1 router and routers R1 and R2. Each router only advertises its Loopback 0 interface into BGP. Example 23-1 shows the commands to achieve this.

Figure 23-9 EBGP Configuration Example Topology



Lo0
.1
SP1
Gi0/1
10.0.3.0/24
.10
.10
BGP AS 65000
Gi0/2
192.168.1.0/24
192.168.2.0/24
R1
Gi0/1
Lo0
.1
.11
10.0.1.0/24
BGP AS 65100
R2
.11
Lo0
.1
Gi0/2
10.0.2.0/24
BGP AS 65200

Example 23-1 Configuring EBGP on SP1, R1, and R2

```
SP1
router bgp 65000
 neighbor 192.168.1.11 remote-as 65100
 neighbor 192.168.2.11 remote-as 65200
 network 10.0.3.0 mask 255.255.255.0
```

```
R1
router bgp 65100
 neighbor 192.168.1.10 remote-as 65000
 network 10.0.1.0 mask 255.255.255.0
```

```
R2
router bgp 65200
 neighbor 192.168.2.10 remote-as 65000
 network 10.0.2.0 mask 255.255.255.0
```

To enable BGP, you need to start the BGP process by using the **router bgp** *as-number* command in global configuration mode. You can configure only a single BGP AS number on a router. In this example, SP1 belongs to AS 65000, R1 belongs to AS 65100, and R2 belongs to AS 65200. To configure a neighbor relationship, you use the **neighbor** *neighbor-ip-address* **remote-as** *remote-as-number* command in BGP router configuration mode. An external BGP peering session must span a maximum of one hop, by default. If not specified otherwise, the IP addresses for an external BGP session must be directly connected to each other.

To specify the networks to be advertised by the BGP routing process, use the **network** router configuration command. The meaning of the **network** command in BGP is radically different from the meaning of the command in other routing protocols. In all other routing protocols, the **network** command indicates interfaces over which the routing protocol will be run. In BGP, it indicates which routes should be injected into the BGP table on the local router. Also, BGP never runs over individual interfaces; rather, it is run over TCP sessions with manually configured neighbors.

BGP Version 4 (BGP4) is a classless protocol, which means its routing updates include the IP address and the subnet mask. The combination of the IP address and the subnet mask is called an *IP prefix*. An IP prefix can be a subnet, a major network, or a summary.

To advertise networks into BGP, you can use the **network** command with the **mask** keyword and the subnet mask specified. If an exact match is not found in the IP routing table, the network is not advertised.

The **network** command with no **mask** option uses the classful approach to insert a major network into the BGP table. Nevertheless, if you do not also enable automatic summarization, an exact match with the valid route in the routing table is required.

Verifying EBGP

Example 23-2 demonstrates the use of the **show ip bgp summary** command. This command allows you to verify the state of the BGP sessions described in Figure 23-9.

Example 23-2 Verifying an EBGP Session

```
SP1# show ip bgp summary
BGP router identifier 10.0.3.1, local AS number 1
BGP table version is 3, main routing table version 3
2 network entries using 296 bytes of memory
2 path entries using 128 bytes of memory
3/2 BGP path/bestpath attribute entries using 408 bytes of memory
2 BGP AS-PATH entries using 48 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 880 total bytes of memory
BGP activity 5/3 prefixes, 5/3 paths, scan interval 60 secs

Neighbor       V        AS MsgRcvd MsgSent    TblVer  InQ OutQ Up/Down  State/PfxRcd
192.168.1.11   4     65100       5       6         3    0    0 00:00:10        1
192.168.2.11   4     65200       5       6         3    0    0 00:00:10        1
```

The first section of the **show ip bgp summary** command output describes the BGP table and its content:

- It describes the BGP router ID of the router and the local AS number, where the router ID is derived from SP1's loopback interface address.
- The BGP table version is the version number of the local BGP table; this number is increased every time the table is changed.

The second section of the **show ip bgp summary** command output is a table that shows the current neighbor statuses. There is one line of text for each neighbor configured. The information that is displayed is as follows:

- The IP address of the neighbor, which is derived from the configured **neighbor** command
- The BGP version number that the router uses when communicating with the neighbor
- The AS number of the remote neighbor, which is derived from the configured **neighbor** command
- The number of messages and updates that have been received from the neighbor since the session was established
- The number of messages and updates that have been sent to the neighbor since the session was established
- The version number of the local BGP table included in the most recent update to the neighbor
- The number of messages that are waiting to be processed in the incoming queue from this neighbor
- The number of messages that are waiting in the outgoing queue for transmission to the neighbor
- How long the neighbor has been in the current state and the name of the current state (the state Established is not displayed, however, so the lack of a state name indicates Established)
- The number of received prefixes from the neighbor if the current state between the neighbors is Established

In this example, SP1 has established sessions with the following neighbors:

- 192.168.1.11, which is the IP address of R1 and is in AS 65100
- 192.168.2.11, which is the IP address of R2 and is in AS 65200

From each of the neighbors, SP1 has received one prefix (that is, one network).

Example 23-3 displays the use of the **show ip bgp neighbors** command on SP1, which provides further details about each configured neighbor. If the command is entered without specifying a particular neighbor, then all neighbors are provided in the output.

Example 23-3 Verifying EBGP Neighbor Information

```
SP1# show ip bgp neighbors 192.168.1.11
BGP neighbor is 192.168.1.11,  remote AS 65100, external link
  BGP version 4, remote router ID 10.0.1.1
  BGP state = Established, up for 00:01:16
  Last read 00:00:24, last write 00:00:05, hold time is 180, keepalive interval
  is 60 seconds
  Neighbor sessions:
    1 active, is not multisession capable (disabled)
  Neighbor capabilities:
    Route refresh: advertised and received(new)
    Four-octets ASN Capability: advertised and received
    Address family IPv4 Unicast: advertised and received
    Enhanced Refresh Capability: advertised and received
    Multisession Capability:
    Stateful switchover support enabled: NO for session 1
  <... output omitted ...>



SP1# show ip bgp neighbors 192.168.2.11
BGP neighbor is 192.168.2.11,  remote AS 65200, external link
  BGP version 4, remote router ID 10.0.2.1
  BGP state = Established, up for 00:02:31
  Last read 00:00:42, last write 00:00:11, hold time is 180, keepalive interval
  is 60 seconds
  Neighbor sessions:
    1 active, is not multisession capable (disabled)
  Neighbor capabilities:
    Route refresh: advertised and received(new)
    Four-octets ASN Capability: advertised and received
    Address family IPv4 Unicast: advertised and received
    Enhanced Refresh Capability: advertised and received
    Multisession Capability:
    Stateful switchover support enabled: NO for session 1
<... output omitted ...>
```

The designation *external link* indicates that the peering relationship is made via EBGP and that the peer is in a different AS.

The status *active* indicates that the BGP session is attempting to establish a connection with the peer. This state implies that the connection has not yet been established. In this case, the sessions are established between SP1 and its two neighbors R1 and R2.

Notice in the output in Example 23-3 the mention of "Address family IPv4 Unicast" support. Since the release of Multiprotocol BGP (MP-BGP) in RFC 4760, BGP supports multiple address families—IPv4, IPv6, and MPLS VPNv4 and VPNv6—as well as either unicast or multicast traffic. The configuration and verification commands presented here focus on the traditional or legacy way of enabling and verifying BGP on a Cisco router. (MP-BGP configuration and verification are beyond the scope of the ENCOR certification exam objectives, so these topics are not covered in this book.)

Example 23-4 shows the use of the **show ip bgp** command on SP1, which displays the router's BGP table and allows you to verify that the router has received the routes that are being advertised by R1 and R2.

Example 23-4 Verifying the BGP Table

```
SP1# show ip bgp
BGP table version is 4, local router ID is 10.0.3.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

    Network          Next Hop            Metric LocPrf Weight Path
*>  10.0.1.0/24      192.168.1.11             0             0 65100 i
*>  10.0.2.0/24      192.168.2.11             0             0 65200 i
*>  10.0.3.0/24      0.0.0.0                  0         32768 i
```

In Example 23-4, SP1 has the following networks in the BGP table:

- 10.0.3.0/24, which is locally originated via the **network** command on SP1; notice the next hop, 0.0.0.0
- 10.0.1.0/24, which has been announced from the 192.168.1.11 (R1) neighbor
- 10.0.2.0/24, which has been announced from the 192.168.2.11 (R2) neighbor

If the BGP table contains more than one route to the same network, the alternate routes are displayed on successive lines. The BGP path selection process selects one of the available routes to each of the networks as the best. This route is designated by the > character in the left column. Each path in this example is marked as the best path because there is only one path to each of the networks.

The columns Metric, LocPrf, Weight, and Path contain the attributes that BGP uses in determining the best path.

Example 23-5 verifies the routing table on SP1. Routes learned via EBGP are marked with an administrative distance (AD) of 20. The metric 0 reflects the BGP multi-exit discriminator (MED) metric value, which is 0, as shown in Example 23-4.

Example 23-5 Verifying the Routing Table

```
SP1# show ip route
<. . . output omitted . . .>
Gateway of last resort is not set

      10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
B        10.0.1.0/24 [20/0] via 192.168.1.11, 00:20:31
B        10.0.2.0/24 [20/0] via 192.168.2.11, 00:20:14
C        10.0.3.0/24 is directly connected, Loopback0
L        10.0.3.1/32 is directly connected, Loopback0
      192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.1.0/24 is directly connected, GigabitEthernet0/1
L        192.168.1.10/32 is directly connected, GigabitEthernet0/1
      192.168.2.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.2.0/24 is directly connected, GigabitEthernet0/2
L        192.168.2.10/32 is directly connected, GigabitEthernet0/2
```

Both customer networks are in the routing table via BGP, as indicated with the letter B:

- Network 10.0.1.0/24 is the simulated LAN in AS 65100 advertised by R1.
- Network 10.0.1.2.0/24 is the simulated LAN in AS 65200 advertised by R2.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	11, 12
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	7

Day 22

First-Hop Redundancy Protocols

ENCOR 350-401 Exam Topics

Architecture

- Explain the different design principles used in an enterprise network
    - High availability techniques such as redundancy, FHRP, and SSO

Infrastructure

- IP Services
    - Configure first hop redundancy protocols, such as HSRP and VRRP

Key Topics

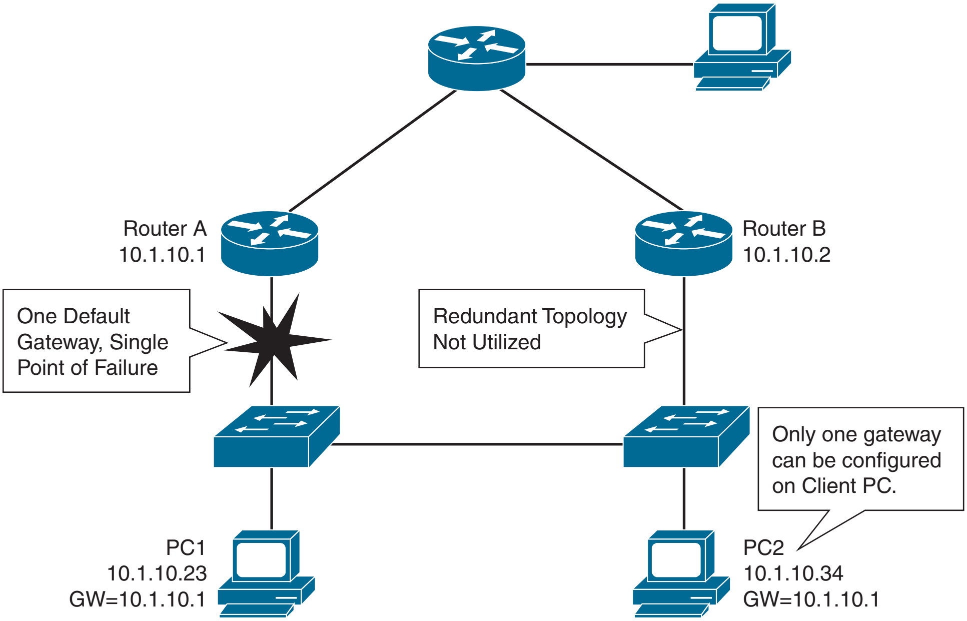
Today we review concepts related to first-hop redundancy protocols (FHRPs). Hosts on an enterprise network have only a single gateway address configured for use when they need to communicate with hosts on a different network. If that gateway fails, hosts are not able to send any traffic to hosts that are not in their own broadcast domain. Building network redundancy at the gateway is a good practice for network reliability. Today we explore network redundancy, including the router redundancy protocols Hot Standby Router Protocol (HSRP) and Virtual Router Redundancy Protocol (VRRP).

Default Gateway Redundancy

When a host determines that a destination IP network is not on its local subnet, it forwards the packet to the default gateway. Although an IP host can run a dynamic routing protocol to build a list of reachable networks, most IP hosts rely on a gateway that is statically configured or a gateway learned using Dynamic Host Configuration Protocol (DHCP).

Having redundant equipment alone does not guarantee uptime. In Figure 22-1, both Router A and Router B are responsible for routing packets for the 10.1.10.0/24 subnet. Because the routers are deployed as a redundant pair, if Router A becomes unavailable, the Interior Gateway Protocol (IGP) can quickly and dynamically converge and determine that Router B should now transfer packets that would otherwise have gone through Router A. Most workstations, servers, and printers, however, do not receive this type of dynamic routing information.

Figure 22-1 Default Gateway Redundancy Example


Router A
10.1.10.1
Router B
10.1.10.2
One Default Gateway, Single Point of Failure
Redundant Topology Not Utilized
Only one gateway can be configured on Client PC.
PC1
10.1.10.23
GW=10.1.10.1
PC2
10.1.10.34
GW=10.1.10.1

Each end device is configured with a single default gateway Internet Protocol (IP) address that does not dynamically update when the network topology changes. If the default gateway fails, the local device is unable to send packets off the local network segment. As a result, the host is isolated from the rest of the network. Even if a redundant router exists that could serve as a default gateway for that segment, there is no dynamic method by which these devices can determine the address of a new default gateway.

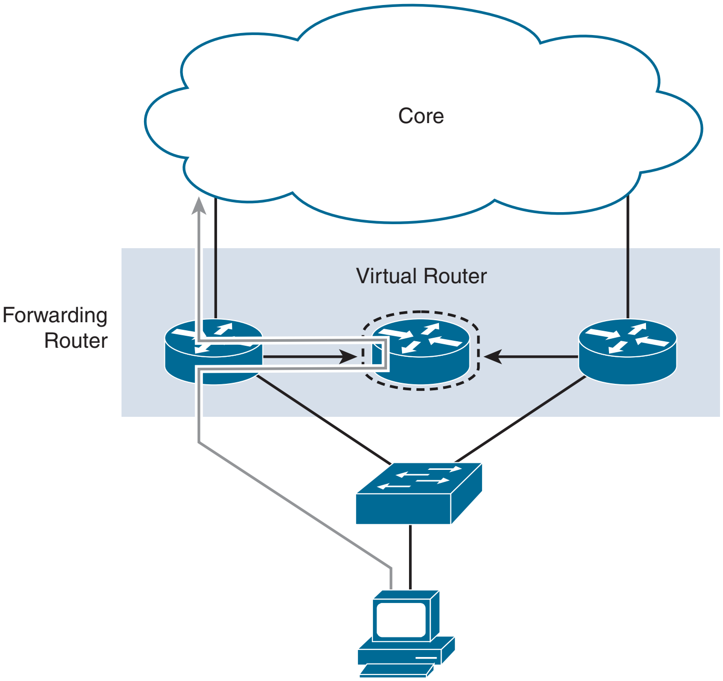
First Hop Redundancy Protocol

Figure 22-2 represents a generic router FHRP with a set of routers working together to present the illusion of a single router to the hosts on the local-area network (LAN). By sharing an IP (Layer 3) address and a Media Access Control (MAC) (Layer 2) address, two or more routers can act as a single "virtual" router.

Hosts that are on the local subnet configure the IP address of the virtual router as their default gateway. When a host needs to communicate to another IP host on a different subnet, it uses Address Resolution Protocol (ARP) to resolve the MAC address of the default gateway. The ARP resolution returns the MAC address of the virtual router. The packets that devices send to the MAC address of the virtual router can then be routed to their destination by any active or standby router that is part of that virtual router group.

You use an FHRP to coordinate two or more routers as the devices that are responsible for processing the packets that are sent to the virtual router. The host devices send traffic to the address of the virtual router. The actual (physical) router that forwards this traffic is transparent to the end stations.

Figure 22-2 FHRP Operations



The redundancy protocol provides the mechanism for determining which router should take the active role in forwarding traffic and determining when a standby router should take over that role. The transition from one forwarding router to another is also transparent to the end devices.

Cisco routers and switches can support three different FHRP technologies:

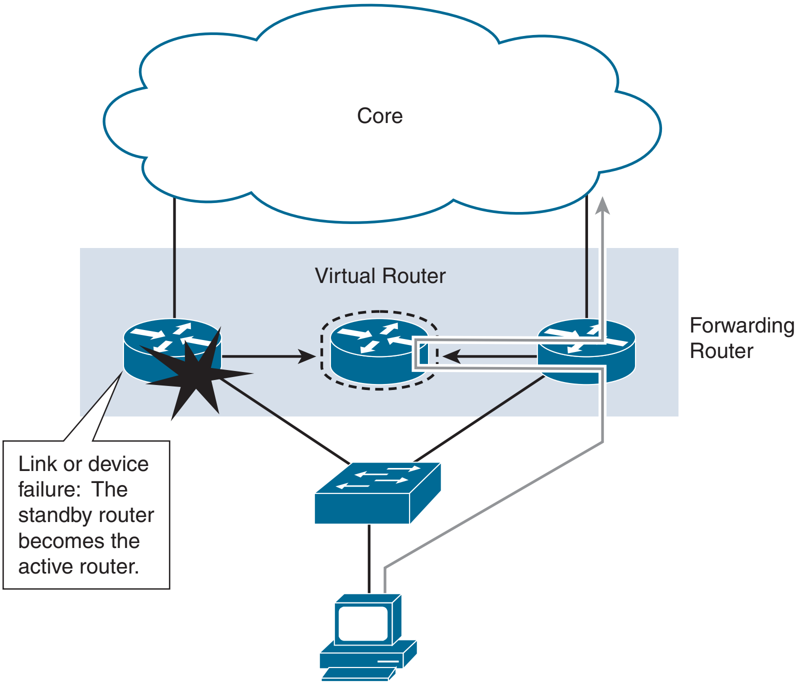
- **Hot Standby Router Protocol (HSRP):** HSRP is an FHRP that Cisco designed to create a redundancy framework between network routers or multilayer switches to achieve default gateway failover capabilities. Only one router forwards traffic. HSRP is defined in RFC 2281.
- **Virtual Router Redundancy Protocol (VRRP):** VRRP is an open FHRP standard that offers the ability to add more than two routers for additional redundancy. Only one router forwards traffic. VRRP is defined in RFC 5798.
- **Gateway Load Balancing Protocol (GLBP):** GLBP is an FHRP that Cisco designed to allow multiple active forwarders to handle load balancing for outgoing traffic. (GLBP is beyond the scope of the ENCOR exam and is therefore not covered in this book.)

A common feature of FHRPs is to provide default gateway failover that is transparent to hosts.

Figure 22-3 illustrates what occurs when the active device or active forwarding link fails:

1. The standby router stops seeing hello messages from the forwarding router.
2. The standby router assumes the role of the forwarding router.
3. Because the new forwarding router assumes both the IP and MAC addresses of the virtual router, the end stations see no disruption in service.

Figure 22-3 FHRP Failover Process


Core
Virtual Router
Forwarding Router
Link or device failure: The standby router becomes the active router.

HSRP

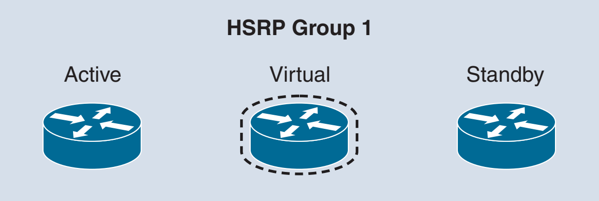
HSRP is a Cisco-proprietary protocol that was developed to allow several multilayer switches or routers to appear as a single gateway IP address. HSRP allows two physical routers to work together in an HSRP group to provide a virtual IP address and an associated virtual MAC address.

The end hosts use the virtual IP address as their default gateway and learn the virtual MAC address via ARP. One of the routers in the group is active and responsible for the virtual addresses. The other router is in a standby state and monitors the active router.

If there is a failure on the active router, the standby router assumes the active state. The virtual addresses are always functional, regardless of which physical router is responsible for them. The end hosts are not aware of any changes in the physical routers.

HSRP defines a standby group of routers, as illustrated in Figure 22-4, with one router that is designated as the active router. HSRP provides gateway redundancy by sharing IP and MAC addresses between redundant gateways. The protocol consists of virtual MAC and IP addresses that two routers that belong to the same HSRP group share with each other.

Figure 22-4 HSRP Standby Group



The HSRP active router has the following characteristics:

- Responds to default gateway ARP requests with the virtual router MAC address
- Assumes active forwarding of packets for the virtual router
- Sends hello messages
- Knows the virtual router IP address

The HSRP standby router has the following characteristics:

- Sends hello messages
- Listens for periodic hello messages
- Knows the virtual IP address
- Assumes active forwarding of packets if it does not hear from the active router

Hosts on the IP subnet that are serviced by HSRP configure their default gateway with the HSRP group virtual IP address. The packets that are received on the virtual IP address are forwarded to the active router.

The function of the HSRP standby router is to monitor the operational status of the HSRP group and to quickly assume the packet-forwarding responsibility if the active router becomes inoperable.

HSRP Group

You assign routers to a common HSRP group by using the following interface configuration command:

```
Router(config-if)# standby group-number ip virtual-ip
```

If you configure HSRP on a multilayer switch, it is a good practice to configure the HSRP group number equal to the VLAN number. Doing so makes troubleshooting easier. HSRP group numbers are locally significant to an interface. For example, HSRP Group 1 on interface VLAN 22 is independent from HSRP Group 1 on interface VLAN 33.

One of the two routers in a group is elected as active and the other will be elected as standby. In an HSRP group with more routers, the other routers are in the listen state. Roles are elected based on the exchange of HSRP hello messages. When the active router fails, the other HSRP routers stop seeing hello messages from the active router. The standby router then assumes the role of the active router. If other routers participate in the group, they contend to be the new standby router. If both the active and standby routers fail, all other routers in the group contend for the active and standby router roles. As the new active router assumes both the IP address and the MAC address of the virtual router, the end stations see no disruption in the service. The end stations continue to send packets to the virtual router MAC address, and the new active router forwards the packets toward their destination.

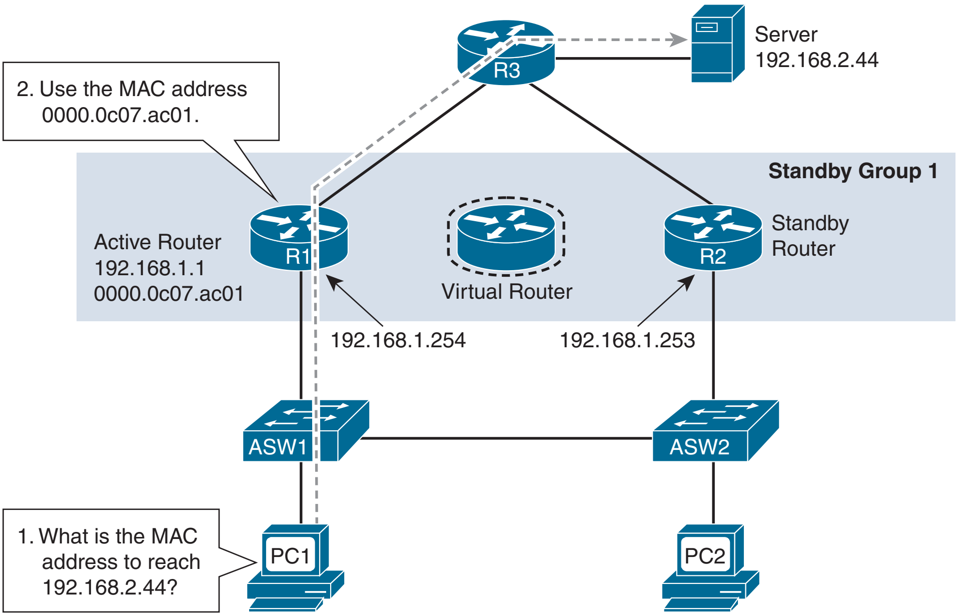
HSRPv1 active and standby routers send hello messages to the multicast address 224.0.0.2, UDP port 1985.

The ICMP protocol allows a router to redirect an end station to send packets for a particular destination to another router on the same subnet—if the first router knows that the other router has a better path to that particular destination. As with default gateways, if the router to which an end station has been redirected for a particular destination fails, the end station packets to that destination are not delivered. In standard HSRP, this action is exactly what happens. For this reason, it is recommended to disable ICMP redirects if HSRP is turned on.

The HSRPv1 virtual MAC address is in the format 0000.0c07.ac*XX*, where *XX* is the HSRP group number, converted from decimal to hexadecimal. Clients use this MAC address to forward data.

Figure 22-5 illustrates what occurs when PC1 tries to reach the server at address 192.168.2.44. In this scenario, the virtual IP address for Standby Group 1 is 192.168.1.1.

Figure 22-5 HSRP Forwarding


Server
192.168.2.44
R3
2. Use the MAC address 0000.0c07.ac01.
Standby Group 1
Active Router
192.168.1.1
0000.0c07.ac01
R1
Virtual Router
R2
Standby
Router
192.168.1.254
192.168.1.253
ASW1
ASW2
1. What is the MAC address to reach 192.168.2.44?
PC1
PC2

If an end station sends a packet to the virtual router MAC address, the active router receives and processes that packet. If an end station sends an ARP request with the virtual router IP address, the active router replies with the virtual router MAC address. In this example, R1 assumes the active role and forwards all frames that are addressed to the well-known MAC address 0000.0c07.ac01.

Whereas ARP and **ping** use the HSRP virtual MAC address, the router responds to **traceroute** with its own MAC address. This is useful in troubleshooting when you need to determine which actual router is used for the traffic flow.

During a failover transition, the newly active router sends three gratuitous ARP requests so that the Layer 2 devices can learn the new port of the virtual MAC address.

HSRP Priority and HSRP Preempt

The HSRP priority is a parameter that enables you to choose the active router between HSRP-enabled devices in a group. The priority is a value between 0 and 255. The default value is 100. The device with the highest priority becomes active.

If HSRP group priorities are the same, the device with the highest IP address becomes active. In the example illustrated in Figure 22-5, R1 is the active router since it has the highest IP address.

Setting priority is wise for deterministic reasons. You want to know how your network will behave under normal conditions. Knowing that R1 is the active gateway for clients in the 192.168.1.0/24 LAN enables you to write good documentation.

Use the following interface configuration command to change the HSRP priority of an interface for a specific group:

```
Router(config-if)# standby group-number priority priority-value
```

Changing the priority of R2 to 110 for standby group 1 does not automatically allow it to become the active router because preemption is not enabled by default. Preemption is the ability of an HSRP-enabled device to trigger the reelection process. You can configure a router to preempt or immediately take over the active role if its priority is the highest at any time. Use the following interface configuration command to change the HSRP priority:

```
Router(config-if)# standby group preempt [delay [minimum seconds] [reload seconds]]
```

By default, after you enter this command, the local router can immediately preempt another router that has the active role. To delay the preemption, use the **delay** keyword followed by one or both of the following parameters:

- Add the **minimum** keyword to force the router to wait for a specified number of seconds (0 to 3600) before attempting to overthrow an active router with a lower priority. This delay time begins as soon as the router is capable of assuming the active role, such as after an interface comes up or after HSRP is configured.
- Add the **reload** keyword to force the router to wait for a specified number of seconds (0 to 3600) after it has been reloaded or restarted. This is useful if there are routing protocols that need time to converge. The local router should not become the active gateway before its routing table is fully populated; if it becomes the active gateway too soon, it might not be capable of routing traffic properly.

Preemption is an important feature of HSRP that allows the primary router to resume the active role when it comes back online after a failure or a maintenance event. Preemption is a desired behavior because it forces a predictable routing path for the LAN traffic during normal operations.

It also ensures that the Layer 3 forwarding path for a LAN parallels the Layer 2 STP forwarding path whenever possible.

When a preempting device is rebooted, HSRP preemption communication should not begin until the router has established full connectivity with the rest of the network. This situation allows the routing protocol convergence to occur more quickly, after the preferred router is in an active state.

To accomplish this setup, measure the system boot time and set the HSRP preemption delay to a value that is about 50% greater than the boot time of the device. This value ensures that the router establishes full connectivity to the network before the HSRP communication occurs.

HSRP Timers

An HSRP hello message contains the priority of the router, the hello time, and the hold time parameter values. The hello timer parameter value indicates the interval of time between the hello messages that the router sends. The hold time parameter value indicates how long the current hello message is considered valid. The **standby timers** command includes an **msec** parameter to allow for subsecond failovers. Lowering the hello timer results in increased traffic for hello messages and should be used cautiously.

If an active router sends a hello message, the receiving routers consider the hello message to be valid for one hold time period. The hold time value should be at least three times the value of the hello time. The hold time value must be greater than the value of the hello time.

You can adjust the HSRP timers to tune the performance of HSRP on distribution devices in order to increase their resilience and reliability in routing packets off the local LAN.

By default, the HSRP hello time is 3 seconds, and the hold time is 10 seconds, which means the failover time could be as much as 10 seconds for clients to start communicating with the new default gateway. Sometimes, this interval may be excessive for application support. The hello time and the hold time parameters are configurable. To configure the time between the hello messages and the time before other group routers declare the active or standby router to be nonfunctioning, enter the following command in interface configuration mode:

```
Router(config-if)# standby group-number timers [msec] hellotime [msec] holdtime
```

The hello interval is specified in seconds (1 to 255) unless the **msec** keyword is used. The dead interval, also specified in seconds (1 to 255), is a time before the active or standby router is declared to be down, unless the **msec** keyword is used.

The hello and dead timer intervals must be identical for all the devices within an HSRP group.

To reinstate the default standby timer values, enter the **no standby** *group-number* **timers** command.

Ideally, to achieve fast convergence, the timer values should be configured to be as low as possible. Within milliseconds after the active router fails, the standby router can detect the failure, expire the hold time interval, and assume the active role.

Nevertheless, the timer configuration should also consider other parameters that are relevant to the network convergence. For example, both HSRP routers may run dynamic routing protocols. The routing protocol probably has no awareness of the HSRP configuration, and it sees both routers as individual hops toward other subnets. If HSRP failover occurs before the dynamic routing protocol converges, suboptimal routing information may still exist. In a worst-case scenario, the dynamic routing protocol continues seeing the failed router as the best next hop to other networks, and packets are lost. When you configure HSRP timers, make sure they harmoniously match the other timers that can influence which path is chosen to carry packets in your network.

HSRP State Transition

An HSRP router can be in one of five states, as illustrated in Table 22-1.

Table 22-1 HSRP States

State	Description
Initial	The state at the start. Also, it is the state after a configuration change or when an interface first comes up.
Listen	The router knows the virtual IP address. It listens for hello messages from other routers.
Speak	The router sends periodic hello messages and actively participates in the election of the active or standby router.
Standby	The router is a candidate to become the next active router and sends periodic hello messages. With the exclusion of transient conditions, there is, at most, one router in the group in standby state.
Active	The router currently forwards packets that are sent to the group virtual MAC address. The router sends periodic hello messages. With the exclusion of transient conditions, there must be, at most, one router in active state in the group.

When a router exists in one of these states, it performs the actions that are required by that state. Not all HSRP routers in the group transition through all states. In an HSRP group with three or more routers, a router that is not the standby or active router remains in the listen state. In other words, no matter how many devices are participating in HSRP, only one device can be in the active state, and one other device can be in the standby state. All other devices are in the listen state.

All routers begin in the initial state. This state is the starting state, and it indicates that HSRP is not running. This state is entered via a configuration change, such as when HSRP is disabled on an interface or when an HSRP-enabled interface is first brought up (such as when the **no shutdown** command is issued).

The purpose of the listen state is to determine if there are any active or standby routers already present in the group. In the speak state, the routers actively participate in the election of the active router, standby router, or both.

HSRP Advanced Features

There are a few options available with HSRP that can allow for more complete insight into network capabilities and add security to the redundancy process. Objects can be tracked, allowing for events other than actual device or HSRP interface failures to trigger state transition. By using Multigroup Hot Standby Routing Protocol (MHSRP), two routers can actively process flows for different standby groups. You can also add security to HSRP by configuring authentication on the protocol.

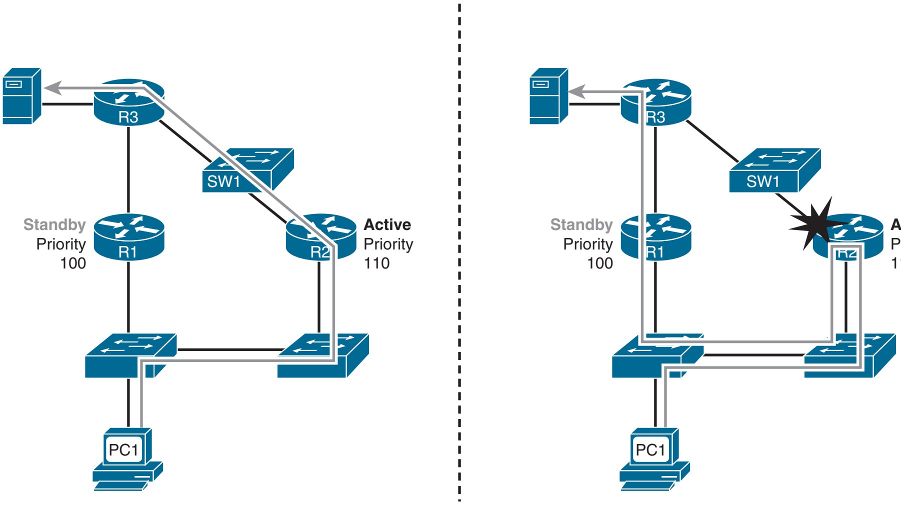
HSRP Object Tracking

HSRP can track objects, and it can decrement priority if an object fails. By default, the HSRP active router loses its status only if the HSRP-enabled interface fails or the HSRP router itself fails. However, it is possible to use object tracking to trigger an HSRP active router election.

When the conditions defined by the object are fulfilled, the router priority remains the same. When the object fails, the router priority is decremented. The amount of decrease can be configured. The default value is 10.

In Figure 22-6, R1 and R2 are configured with HSRP. R2 is configured to be the active default gateway, and R1 will take over if R2 or the HSRP-enabled interface on R2 fails.

Figure 22-6 HSRP with No Interface Tracking

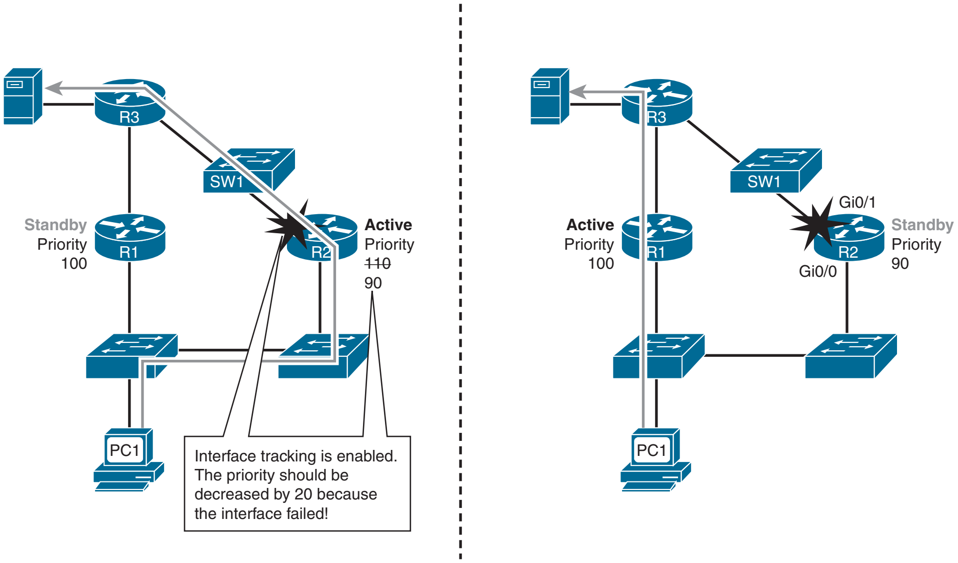


What happens if the R2 uplink fails? The uplink interface is not an HSRP-enabled interface, so its failure does not affect HSRP. R2 is still the active default gateway. All the traffic from PC1 to the server now has to go to R2, and then it gets routed back to R1 and forwarded to the server; this is an inefficient traffic path.

HSRP provides a solution to this problem: HSRP object tracking. Object tracking allows you to specify another interface on the router for the HSRP process to monitor to alter the HSRP priority for a given group. If the line protocol for the specified interface goes down, the HSRP priority of this router is reduced, allowing another HSRP router with a higher priority to become active. Preemption must be enabled on both routers for this feature to work correctly.

Consider the same scenario as before. In Figure 22-7, the R2 uplink interface fails, but this time HSRP, by virtue of HSRP object tracking, detects this failure, and the HSRP priority for R2 is decreased by 20. With preemption enabled, R1 then takes over as the active HSRP peer because it has a higher priority.

Figure 22-7 HSRP with Interface Object Tracking



Interface tracking is enabled. The priority should be decreased by 20 because the interface failed!

Configuring interface object tracking for HSRP is a two-step process:

1. Define the tracking object criteria by using the global configuration command **track** *object-number* **interface** *interface-id* **line-protocol**.
2. Associate the object with a specific HSRP group by using the **standby** *group-number* **track** *object-id* **decrement** *decrement-value*.

Example 22-1 shows the commands used on R1 and R2 in Figure 22-7 to configure interface object tracking for HSRP Standby Group 1. Interface GigabitEthernet 0/0 is the HSRP-enabled interface, and interface GigabitEthernet 0/1 is the tracked interface. Preemption is enabled on the HSRP-enabled interface on R1, which allows it to become the new active router when R2's GigabitEthernet 0/1 interface fails. If and when the GigabitEthernet 0/1 interface is repaired, R2 can reclaim the active status, thanks to the preemption feature because its priority returns to 110.

Example 22-1 Configuring Object Tracking for HSRP

```
R2(config)# track 10 interface GigabitEthernet 0/1 line-protocol
R2(config)# interface GigabitEthernet 0/0
R2(config-if)# standby 1 priority 110
R2(config-if)# standby 1 track 10 decrement 20
R2(config-if)# standby 1 preempt
```

```
R1(config)# interface GigabitEthernet 0/0
R1(config-if)# standby 1 preempt
```

You can apply multiple tracking statements to an interface. This may be useful, for example, if the currently active HSRP interface will relinquish its status only upon the failure of two (or more) tracked interfaces.

Beside interfaces, it is possible to also track the presence of routes in the routing table, as well as the status of an IP SLA. A tracked IP route object is considered up and reachable when a routing table entry exists for the route and the route is accessible. To provide a common interface to tracking clients, route metric values are normalized to the range of 0 to 255, where 0 is connected, and 255 is inaccessible. You can track route reachability or even metric values to determine best-path values to the target network. The tracking process uses a per-protocol configurable resolution value to convert the real metric to the scaled metric. The metric value that is communicated to clients is always such that a lower metric value is better than a higher metric value. Use the **track** *object-number* **ip route** *route/prefix-length* **reachability** command to track a route in the routing table.

For an IP SLA, besides tracking the operational state, it is possible to track advanced parameters such as IP reachability, delay, or jitter. Use the **track** *object-number* **ip sla** *operation-number* [**state** | **reachability**] command to track an IP SLA.

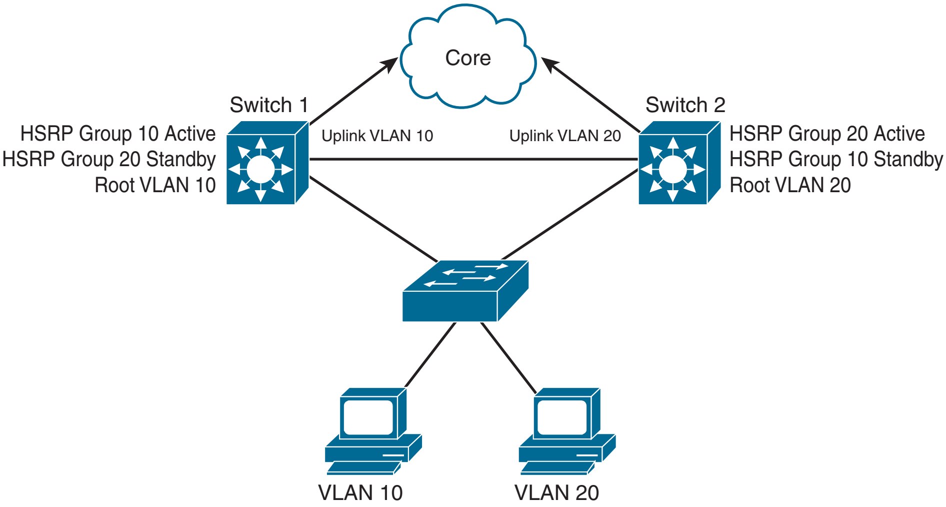
Use the **show track** *object-number* command to verify the state of the tracked interface and use the **show standby** command to verify that tracking is configured.

HSRP Multigroup

HSRP does not support load sharing as part of the protocol specification. However, load sharing can be achieved through the configuration of MHSRP.

In Figure 22-8, two HSRP-enabled multilayer switches participate in two separate VLANs, using IEEE 802.1Q trunks. If you leave the default HSRP priority values, a single multilayer switch will likely become an active gateway for both VLANs, effectively utilizing only one uplink toward the core of the network.

Figure 22-8 HSRP Load Balancing with MHSRP



To use both paths toward the core network, you can configure HSRP with MHSRP. Group 10 is configured for VLAN 10. Group 20 is configured for VLAN 20. For group 10, Switch1 is configured with a higher priority to become the active gateway, and Switch2 becomes the standby gateway. For group 20, Switch2 is configured with a higher priority to become the active gateway, and Switch1 becomes the standby router. Now both uplinks toward the core are utilized: one with VLAN 10 and one with VLAN 20 traffic.

Example 22-2 shows the commands to configure MHSRP on Switch1 and Switch2 in Figure 22-8. Switch1 has two HSRP groups that are configured for two VLANs and that correspond to the STP root configuration. Switch1 is the active router for HSRP group 10 and is the standby router for group 20. Switch2's configuration mirrors the configuration on Switch1.

Example 22-2 Configuring MHSRP

```
Switch1(config)# spanning-tree vlan 10 root primary
Switch1(config)# spanning-tree vlan 20 root secondary
Switch1(config)# interface vlan 10
Switch1(config-if)# ip address 10.1.10.2 255.255.255.0
Switch1(config-if)# standby 10 ip 10.1.10.1
Switch1(config-if)# standby 10 priority 110
Switch1(config-if)# standby 10 preempt
Switch1(config-if)# exit
Switch1(config)# interface vlan 20
Switch1(config-if)# ip address 10.1.20.2 255.255.255.0
Switch1(config-if)# standby 20 ip 10.1.20.1
Switch1(config-if)# standby 20 priority 90
Switch1(config-if)# standby 20 preempt
```

```
Switch2(config)# spanning-tree vlan 10 root secondary
Switch2(config)# spanning-tree vlan 20 root primary
Switch2(config)# interface vlan 10
Switch2(config-if)# ip address 10.1.10.3 255.255.255.0
Switch2(config-if)# standby 10 ip 10.1.10.1
Switch2(config-if)# standby 10 priority 90
Switch2(config-if)# standby 10 preempt
Switch2(config-if)# exit
Switch2(config)# interface vlan 20
Switch2(config-if)# ip address 10.1.20.3 255.255.255.0
Switch2(config-if)# standby 20 ip 10.1.20.1
Switch2(config-if)# standby 20 priority 110
Switch2(config-if)# standby 20 preempt
```

HSRP Authentication

HSRP authentication prevents rogue Layer 3 devices on the network from joining the HSRP group.

A rogue device may claim the active role and can prevent the hosts from communicating with the rest of the network, creating a DoS attack. A rogue router could also forward all traffic and capture traffic from the hosts, achieving a man-in-the-middle attack.

HSRP provides two types of authentication: plaintext and MD5.

To configure plaintext authentication, use the following interface configuration command on HSRP peers:

```
Router(config-if)# standby group-number authentication string
```

With plaintext authentication, a message that matches the key that is configured on an HSRP peer is accepted. The maximum length of a key string is eight characters. Plaintext messages can easily be intercepted, so avoid plaintext authentication if MD5 authentication is available.

To configure MD5 authentication, use the following interface configuration command on HSRP peers:

```
Router(config-if)# standby group-number authentication md5 [key-chain key-chain |
  key-string key-string]
```

Using MD5, a hash is computed on a portion of each HSRP message. The hash is sent along with the HSRP message. When a peer receives the message and a hash, it performs hashing on the received message. If the received hash and the newly computed hash match, the message is accepted. It is very difficult to reverse the hash value itself, and hash keys are never exchanged. MD5 authentication is preferred.

Instead of using a single MD5 key, you can define MD5 strings as keys on a keychain. This method is flexible because it means you can define multiple keys with different validity times.

HSRP Versions

There are two HSRP versions available on most Cisco routers and multilayer switches: HSRPv1 and HSRPv2. Table 22-2 compares the two versions.

Table 22-2 HSRP Versions

HSRPv1	HSRPv2
IPv4	IPv4/IPv6
Group numbers 0–255	Group numbers 0–4095
Virtual MAC address: 0000:0c07:ac*XX* (where *XX* is the HSRP group, in hexadecimal)	Virtual MAC address: 0000:0c9f:f*XXX* (where *XXX* is the HSRP group, in hexadecimal)
Multicast address: 224.0.0.2	Multicast address: 224.0.0.102
Default version	

To enable HSRPv2 on all devices, use the following command in interface configuration mode:

```
Router(config-if)# standby version 2
```

HSRPv1 is the default version on Cisco IOS devices. HSRPv2 is supported in Cisco IOS Software Release 12.2(46)SE and later. HSRPv2 allows group numbers up to 4095, thus allowing you to use the VLAN number as the group number.

HSRPv2 must be enabled on an interface before HSRP for IPv6 can be configured.

HSRPv2 does not interoperate with HSRPv1. All devices in an HSRP group must have the same version configured; otherwise, the hello messages are not understood. An interface cannot operate both HSRPv1 and HSRPv2 because they are mutually exclusive.

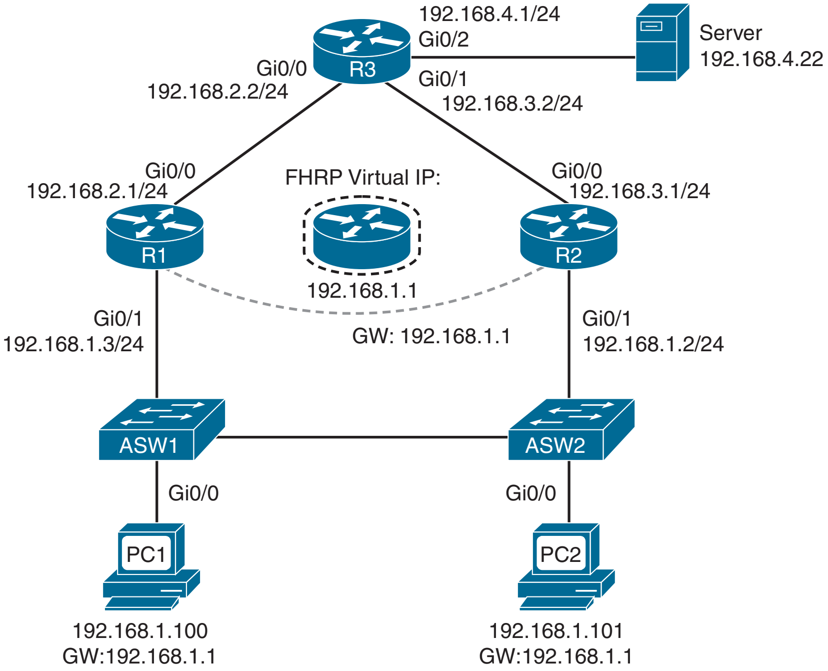
The MAC address of the virtual router and the multicast address for the hello messages are different with HSRPv2. HSRPv2 uses the new IP multicast address 224.0.0.102 to send the hello packets, whereas HSRPv1 uses the multicast address 224.0.0.2. This new address allows Cisco Group Management Protocol (CGMP) multicast processing to be enabled at the same time as HSRP.

HSRPv2 has a different packet format from HSRPv1. It includes a 6-byte identifier field that is used to uniquely identify the sender of the message by its interface MAC address, which makes troubleshooting easier.

HSRP Configuration Example

Figure 22-9 shows a topology in which R1 and R2 are gateway devices available for PCs in the 192.168.1.0/24 subnet. R1 is configured to become the HSRP active router, and R2 is the HSRP standby router. R1 is configured for object tracking so that it can track the status of its GigabitEthernet 0/0 interface. If the interface fails, R2 should become the HSRP active router.

Figure 22-9 HSRP Configuration Example


192.168.4.1/24
Gi0/2
Server
192.168.4.22
Gi0/0
R3
192.168.2.2/24
Gi0/1
192.168.3.2/24
Gi0/0
192.168.2.1/24
FHRP Virtual IP:
Gi0/0
192.168.3.1/24
R1
R2
192.168.1.1
Gi0/1
192.168.1.3/24
GW: 192.168.1.1
Gi0/1
192.168.1.2/24
ASW1
ASW2
Gi0/0
Gi0/0
PC1
PC2
192.168.1.100
GW:192.168.1.1
192.168.1.101
GW:192.168.1.1

Example 22-3 shows a complete HSRP configuration, including the use of HSRPv2, object tracking, authentication, timer adjustment, and preemption delay.

Example 22-3 Configuring HSRP

```
R1(config)# track 5 interface GigabitEthernet0/0 line-protocol
R1(config)# interface GigabitEthernet 0/1
R1(config-if)# standby version 2
R1(config-if)# standby 1 ip 192.168.1.1
R1(config-if)# standby 1 priority 110
R1(config-if)# standby 1 authentication md5 key-string 31DAYS
R1(config-if)# standby 1 timers msec 200 msec 750
R1(config-if)# standby 1 preempt delay minimum 300
R1(config-if)# standby 1 track 5 decrement 20
```

```
R2(config)# interface GigabitEthernet 0/1
R2(config-if)# standby version 2
R2(config-if)# standby 1 ip 192.168.1.1
R2(config-if)# standby 1 authentication md5 key-string 31DAYS
R2(config-if)# standby 1 timers msec 200 msec 750
R2(config-if)# standby 1 preempt
```

R2 is not configured with object tracking because it will become active only if R1 reports a lower priority. Also, notice the preemption delay configured on R1. This gives R1 time to fully converge with the network before reclaiming the active status when GigabitEthernet 0/0 is repaired. No preemption delay is configured on R2 because it needs to immediately claim the active status once R1's priority drops below 100.

Example 22-4 shows the use of the HSGHS verification commands **show track**, **show standby brief**, and **show standby**.

Example 22-4 Verifying Object Tracking and HSRP

```
R1# show track
Track 5
  Interface GigabitEthernet0/0 line-protocol
  Line protocol is Up
    1 change, last change 00:01:08
```

```
R1# show standby
GigabitEthernet0/1 - Group 1 (version 2)
  State is Active
    2 state changes, last state change 00:03:16
  Virtual IP address is 192.168.1.1
  Active virtual MAC address is 0000.0c9f.f001
    Local virtual MAC address is 0000.0c9f.f001 (v2 default)
  Hello time 200 msec, hold time 750 msec
    Next hello sent in 0.064 secs
  Authentication MD5, key-string
  Preemption enabled, delay min 300 secs
  Active router is local
  Standby router is 192.168.1.2, priority 100 (expires in 0.848 sec)
  Priority 110 (configured 110)
    Track object 5 state Up decrement 20
  Group name is "hsrp-Et0/1-1" (default)

R1# show standby brief
                      P indicates configured to preempt.
                      |
Interface    Grp  Pri P State   Active          Standby         Virtual IP
Gi0/1        1    110 P Active  local           192.168.1.2     192.168.1.1
```

The **show track** command output confirms that GigabitEthernet 0/0 is currently operational. The **show standby** command confirms that HSRPv2 is enabled, that its current state is active, while R2 is standby. The output also confirms that MD5 authentication and preemption are enabled. Finally, notice that the tracking object is currently up but that it decrements the priority by a value of 20 if the tracking object fails.

The **show standby brief** command provides a snapshot of the HSRP status on R1's GigabitEthernet 0/1 interface.

VRRP

VRRP is similar to HSRP, both in operation and in configuration. The VRRP master is analogous to the HSRP active gateway, and the VRRP backup is analogous to the HSRP standby gateway. A VRRP group has one master device and one or multiple backup devices. A device with the highest priority is the elected master. The priority can be a number between 0 and 255. The priority value 0 has a special meaning: It indicates that the current master has stopped participating in VRRP. This setting is used to trigger backup devices to quickly transition to master without having to wait for the current master to time out.

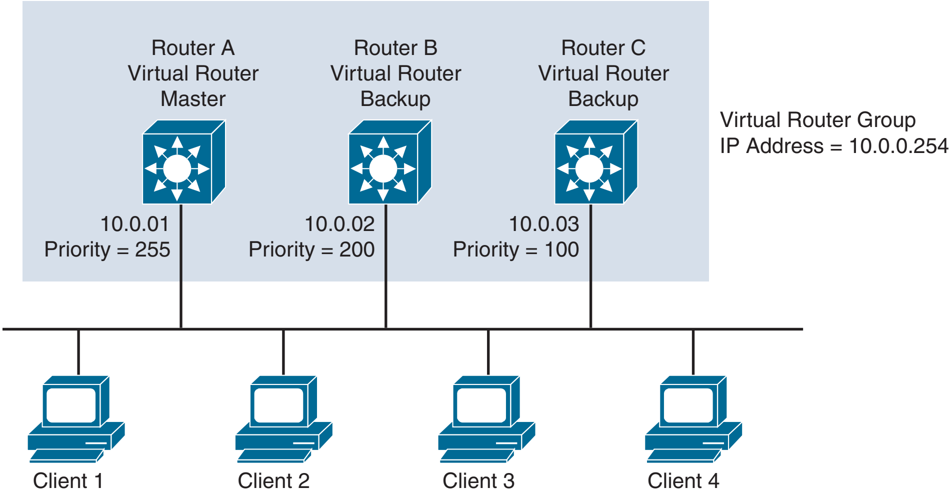
VRRP differs from HSRP in that it allows you to use an address of one of the physical VRRP group members as a virtual IP address. In this case, the device with the used physical address is a VRRP master whenever it is available.

The master is the only device that sends advertisements (analogous to HSRP hellos). Advertisements are sent to the 224.0.0.18 multicast address, with protocol number 112. The default advertisement interval is 1 second. The default hold time is 3 seconds. HSRP, in comparison, has the default hello timer set to 3 seconds and the hold timer set to 10 seconds. VRRP uses the MAC address format 0000.5e00.01*XX*, where *XX* is the group number, in hexadecimal.

Cisco devices allow you to configure VRRP with millisecond timers. You need to manually configure the millisecond timer values on both the master and backup devices. Use the millisecond timers only when absolutely necessary and with careful consideration and testing. Millisecond values work only under favorable circumstances, and you must be aware that the use of the millisecond timer values restricts VRRP operation to Cisco devices only.

In Figure 22-10, Router A, Router B, and Router C are configured as VRRP virtual routers and are members of the same VRRP group. Because Router A has the highest priority, it is elected the master for this VRRP group. End-user devices use it as their default gateway. Routers B and C function as virtual router backups. If the master fails, the device with the highest configured priority becomes the master and provides uninterrupted service for the LAN hosts. When Router A recovers and with preemption enabled, Router A becomes the master again. Unlike with HSRP, with VRRP, preemption is enabled by default.

Figure 22-10 VRRP Example



Load sharing is also available with VRRP and, as with HSRP, multiple virtual router groups can be configured. For instance, in Figure 22-10, you could configure Clients 3 and 4 to use a different default gateway than Clients 1 and 2 use. Then you would configure the three multilayer switches with another VRRP group and designate Router B to be the master VRRP device for the second group.

RFC 5798 defines VRRP support for both IPv4 and IPv6. The default VRRP version on Cisco devices is VRRPv2, and it only supports IPv4. To support both IPv4 and IPv6, you need to enable VRRPv3 by using the global configuration command **fhrp version vrrp v3**. Also, the configuration framework for VRRPv2 and VRRPv3 differs significantly. Legacy VRRPv2 is non-hierarchical in its configuration, while VRRPv3 uses the address family framework. To enter the VRRP address family configuration framework, enter the **vrrp** *group-number* **address-family** [**ipv4** | **ipv6**] interface command.

Like HSRP, VRRP supports object tracking for interface state, IP route reachability, IP SLA state, IP SLA reachability, and so on.

VRRP Authentication

According to RFC 5798, operational experience and further analysis determined that VRRP authentication did not provide sufficient security to overcome the vulnerability of misconfigured secrets, and multiple masters could be elected. Due to the nature of VRRP, even cryptographically protecting VRRP messages does not prevent hostile nodes from behaving as if they are the VRRP master and creating multiple masters. Authentication of VRRP messages could prevent a hostile node from causing all properly functioning routers from going into the backup state. However, having multiple masters can cause as much disruption as having no routers, and authentication cannot prevent this. Also, even if a hostile node cannot disrupt VRRP, it can disrupt ARP and create the same effect as having all routers go into the backup state.

Independent of any authentication type, VRRP includes a mechanism (setting Time to Live [TTL] = 255, checking on receipt) that protects against VRRP packets being injected from another remote network. The TTL setting limits most vulnerability to local attacks.

With Cisco IOS devices, the default VRRPv2 authentication is plaintext. MD5 authentication can be configured by specifying a key string or, as with HSRP, reference to a keychain. Use the **vrrp** *group-number* **authentication text** *key-string* command for plaintext authentication, and use the **vrrp** *group-number* **authentication md5** [**key-chain** *key-chain* | **key-string** *key-string*] command for MD5 authentication.

VRRP Configuration Example

Using the topology in Figure 22-9, Example 22-5 shows the configuration of legacy VRRPv2, and Example 22-6 shows the configuration for address family VRRPv3. R1 is configured as the VRRP master, and R2 is configured as the VRRP backup. Both examples also demonstrate the use of the priority and track features.

Example 22-5 Configuring Legacy VRRPv2

```
R1(config)# track 5 interface GigabitEthernet0/0 line-protocol
R1(config)# interface GigabitEthernet 0/1
R1(config-if)# vrrp 1 ip 192.168.1.1
R1(config-if)# vrrp 1 priority 110
R1(config-if)# vrrp 1 authentication md5 key-string 31DAYS
R1(config-if)# vrrp 1 preempt delay minimum 300
R1(config-if)# vrrp 1 track 5 decrement 20
```

```
R2(config)# interface GigabitEthernet 0/1
R2(config-if)# vrrp 1 ip 192.168.1.1
R2(config-if)# vrrp 1 authentication md5 key-string 31DAYS
```

In Example 22-5, notice that the legacy VRRP syntax is practically identical to the HSRP syntax. Recall that preemption is enabled by default in VRRP.

Example 22-6 Configuring Address Family VRRPv3

```
R1(config)# track 5 interface GigabitEthernet0/0 line-protocol
R1(config)# fhrp version vrrp 3
R1(config)# interface GigabitEthernet 0/1
R1(config-if)# vrrp 1 address-family ipv4
R1(config-if-vrrp)# address 192.168.1.1
R1(config-if-vrrp)# priority 110
R1(config-if-vrrp)# preempt delay minimum 300
R1(config-if-vrrp)# track 5 decrement 20
```

```
R2(config)# fhrp version vrrp 3
R2(config)# interface GigabitEthernet 0/1
R2(config-if)# vrrp 1 address-family ipv4
R2(config-if-vrrp)# address 192.168.1.1
```

In Example 22-6, in the VRRP address family configuration framework, the commands are similar to those used in Example 22-5 except that they are entered hierarchically under the appropriate address families. All VRRP parameters and options are entered under the VRRP instance. Notice that authentication is not supported. Also, it is possible to use VRRPv2 with the address family framework. Use the **vrrpv2** command under the VRRP instance to achieve this.

To verify the operational state of VRRP, use the **show vrrp brief** and **show vrrp** commands, as illustrated in Example 22-7. The output format is similar to what you saw earlier with HSRP. The first part of the example displays the output when using legacy VRRPv2. The second part displays the output when using address family VRRPv3.

Example 22-7 Verifying Legacy VRRPv2 and Address Family VRRPv3

```
! Legacy VRRPv2
R1# show vrrp brief
Interface          Grp  Pri Time  Own Pre  State    Master addr      Group addr
Gi0/1              1    110 3570       Y   Master   192.168.1.3      192.168.1.1
!
R1# show vrrp
Ethernet0/1 - Group 1
  State is Master
  Virtual IP address is 192.168.1.1
  Virtual MAC address is 0000.5e00.0101
  Advertisement interval is 1.000 sec
  Preemption enabled, delay min 300 secs
  Priority is 110
    Track object 5 state UP decrement 20
  Master Router is 192.168.1.3 (local), priority is 110
  Master Advertisement interval is 1.000 sec
  Master Down interval is 3.609 sec (expires in 3.049 sec)
```

```
! Address Family VRRPv3
R1# show vrrp brief
  Interface      Grp  A-F Pri  Time Own Pre State  Master addr/Group addr
  Gi0/1            1 IPv4 110     0  N   Y  MASTER 192.168.1.3 (local) 192.168.1.1
!
R1# show vrrp

GigabitEthernet0/1 - Group 1 - Address-Family IPv4
  State is MASTER
  State duration 2 mins 14.741 secs
  Virtual IP address is 192.168.1.1
  Virtual MAC address is 0000.5E00.0114
  Advertisement interval is 1000 msec
  Preemption enabled, delay min 300 secs (0 msec remaining)
  Priority is 110
    Track object 5 state UP decrement 20
  Master Router is 192.168.1.3 (local), priority is 110
  Master Advertisement interval is 1000 msec (expires in 292 msec)
  Master Down interval is unknown
```

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	**Module, Chapter, or Link**
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	15
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	8

This page intentionally left blank

Day 21

Network Services

ENCOR 350-401 Exam Topics

Infrastructure

- IP Services
  - Describe Network Time Protocol (NTP)
  - Configure and verify NAT/PAT

Key Topics

Today we review two important network services: Network Address Translation (NAT) and Network Time Protocol (NTP). Because public IPv4 addresses are in such high demand but are subject to limited availability, many organizations use private IP addresses internally and use NAT to access public resources. Today we explore the advantages and disadvantages of using NAT and look at the different ways it can be implemented.

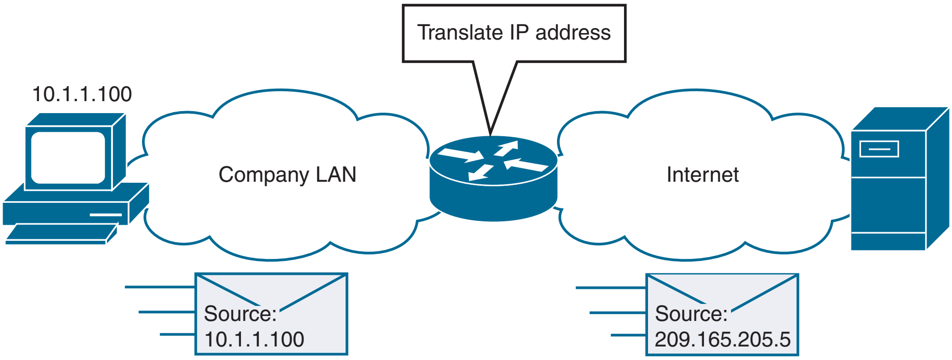
NTP is designed to synchronize the time on a network of machines. From a troubleshooting perspective, it is very important that all the network devices be synchronized to have the correct timestamps in their logged messages. The current protocol is Version 4 (NTPv4), which is documented in RFC 5905. It is backward compatible with Version 3, specified in RFC 1305.

Network Address Translation

Small to medium-sized networks are commonly implemented using private IP addressing, as defined in RFC 1918. Private addressing gives enterprises considerable flexibility in network design. This addressing enables operationally and administratively convenient addressing schemes and easier growth. However, you cannot route private addresses over the Internet. Therefore, network administrators need a mechanism to translate private addresses to public addresses (and back) at the edge of their network. Network Address Translation (NAT), illustrated in Figure 21-1, is a method to do this.

NAT allows users with private addresses to access the Internet by sharing one or more public IP addresses. NAT, which is typically used at the edge of an organization's network where it is connected to the Internet, translates the private addresses of the internal network to publicly registered addresses. You can configure NAT to advertise only one address for an entire network to the outside world. Advertising only one address effectively hides the internal network, providing additional security as a side benefit.

Figure 21-1 NAT Process



However, the NAT process of swapping one address for another is separate from the convention that is used to determine what is public and private, and devices must be configured to recognize which IP networks should be translated. Therefore, NAT can also be deployed internally when there is a clash of private IP addresses, such as when two companies using the same private addressing scheme merge or to isolate different operational units within an enterprise network.

The benefits of NAT include the following:

- NAT eliminates the need to readdress all hosts that require external access, saving time and money.
- NAT conserves addresses through application port-level multiplexing. With Port Address Translation (PAT), which is one way to implement NAT, multiple internal hosts can share a single registered IPv4 address for all external communication. In this type of configuration, relatively few external addresses are required to support many internal hosts. This characteristic conserves IPv4 addresses.
- NAT provides a level of network security. Because private networks do not advertise their addresses or internal topology, they remain reasonably secure when they gain controlled external access with NAT.

The disadvantages of NAT include the following:

- Many IP addresses and applications depend on end-to-end functionality, with unmodified packets forwarded from the source to the destination. By changing end-to-end addresses, NAT blocks some applications that use IP addressing. For example, some security applications, such as digital signatures, fail when the source IP address changes. Applications that use physical addresses instead of a qualified domain name do not reach destinations that are translated across the NAT router. Sometimes, you can avoid this problem by implementing static NAT mappings or using proxy endpoints or servers.
- End-to-end IP traceability is also lost. It becomes much more difficult to trace packets that undergo numerous packet address changes over multiple NAT hops, so troubleshooting is challenging. On the other hand, hackers who want to determine the source of a packet find it difficult to trace or obtain the original source or destination address.

- Using NAT also complicates tunneling protocols, such as IPsec, because NAT modifies the values in the headers. This behavior interferes with the integrity checks that IPsec and other tunneling protocols perform.

- Services that require the initiation of TCP connections from the outside network, or stateless protocols such as those using UDP, can be disrupted. Unless a NAT router makes a specific effort to support such protocols, incoming packets cannot reach their destination. Some protocols can accommodate one instance of NAT between participating hosts (passive mode FTP, for example) but fail when NAT separates both systems from the Internet.

- NAT increases switching delays because translation of each IP address within the packet headers takes time. The first packet is process switched. The router must look at each packet to decide whether it needs to be translated. The router needs to alter the IP header and possibly alter the TCP or UDP header.

NAT Address Types

Cisco defines a number of terms related to NAT:

- **Inside local address:** The IP address assigned to a host on the inside network. This is the address configured as a parameter of the computer OS or received via dynamic address allocation protocols such as DHCP. The IP ranges here are typically from the private IP address ranges described in RFC 1918 and are the addresses that are to be translated:
    - 10.0.0.0/8
    - 172.16.0.0/24
    - 192.168.0.0/16

- **Inside global address:** The address that an inside local address is translated into. This address is typically a legitimate public IP address assigned by the service provider.

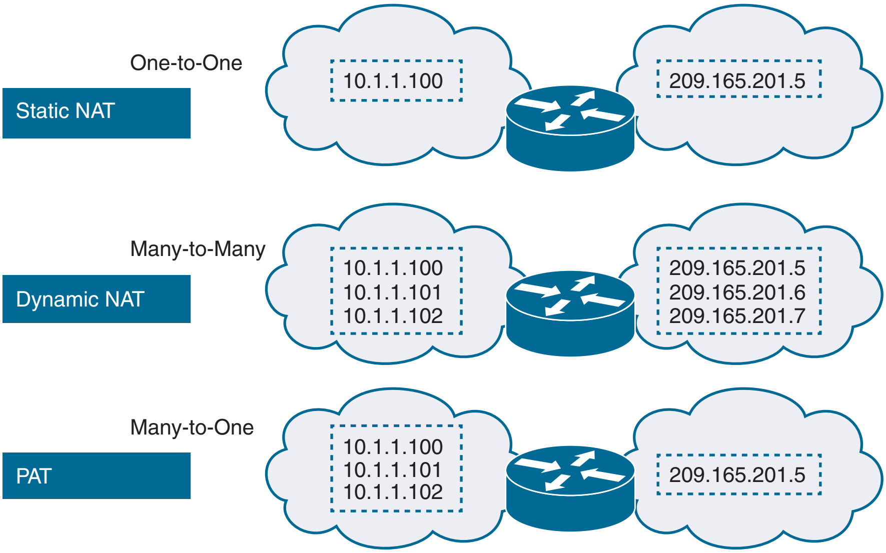
- **Outside global address:** The IPv4 address of a host on the outside network. The outside global address is typically allocated from a globally routable address or network space.

- **Outside local address:** The IPv4 address of an outside host as it appears to its own inside network. The outside local address, which is not necessarily public, is allocated from a routable address space on the inside. This address is typically important when NAT is used between networks with overlapping private addresses as when two companies merge. In most cases, the inside global and outside global addresses are the same, and they indicate the destination address of outbound traffic from a source that is being translated.

A good way to remember what is local and what is global is to add the word *visible*. An address that is locally visible normally implies a private IP address, and an address that is globally visible normally implies a public IP address. *Inside* means internal to your network, and *outside* means external to your network. So, for example, an inside global address means that the device is physically inside your network and has an address that is visible from the Internet.

Figure 21-2 illustrates a topology in which two inside hosts using private RFC 1918 addresses are communicating with the Internet. The router is translating the inside local addresses to inside global addresses that can be routed across the Internet.

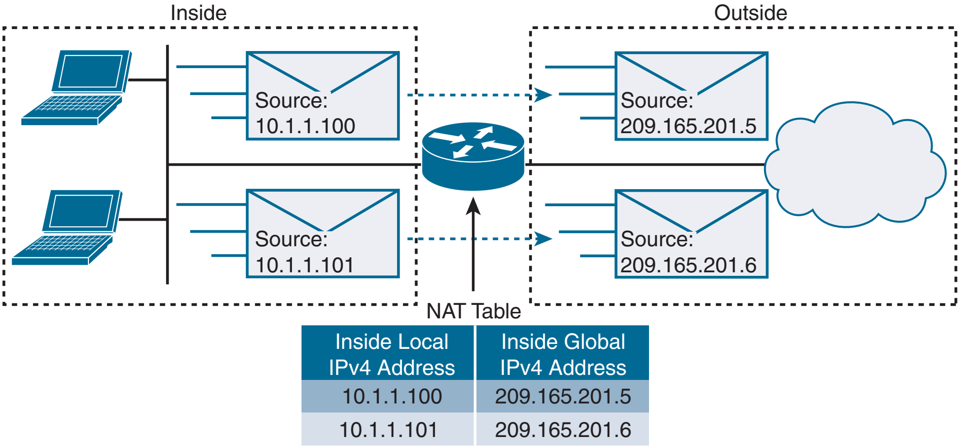
Figure 21-2 NAT Address Types


One-to-One
Static NAT
10.1.1.100
209.165.201.5
Many-to-Many
Dynamic NAT
10.1.1.100
10.1.1.101
10.1.1.102
209.165.201.5
209.165.201.6
209.165.201.7
Many-to-One
PAT
10.1.1.100
10.1.1.101
10.1.1.102
209.165.201.5

NAT Implementation Options

On a Cisco IOS router, NAT can be implemented in three different ways, each of which has clear use cases (see Figure 21-3):

Figure 21-3 NAT Deployment Options


Inside
Outside
Source:
10.1.1.100
Source:
10.1.1.101
Source:
209.165.201.5
Source:
209.165.201.6
NAT Table

Inside Local IPv4 Address	Inside Global IPv4 Address
10.1.1.100	209.165.201.5
10.1.1.101	209.165.201.6

- **Static NAT:** Maps a private IPv4 address to a public IPv4 address (one to one). Static NAT is particularly useful when a device must be accessible from outside the network. This type of NAT is used when a company has a server that needs a static public IP address, such as a web server.

- **Dynamic NAT:** Maps a private IPv4 address to one of many available addresses in a group or pool of public IPv4 addresses. This type of NAT is used, for example, when two companies that are using the same private address space merge. With the use of dynamic NAT readdressing, migrating the entire address space is avoided or at least postponed.

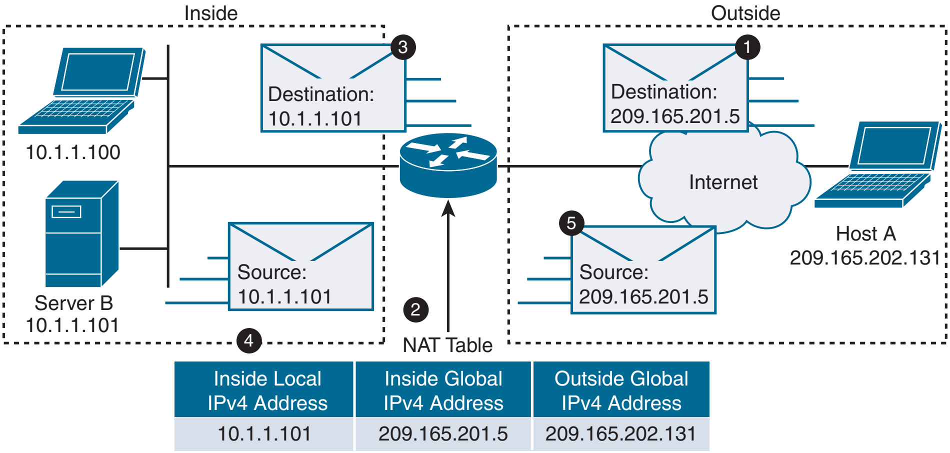
- **PAT:** Maps multiple private IPv4 addresses to a single public IPv4 address (many to one) by using different ports. PAT is also known as NAT overloading. It is a form of dynamic NAT and is the most common use of NAT. It is used every day in your place of business or your home. Multiple users of PCs, tablets, and phones are able to access the Internet, even though only one public IP address is available for that LAN.

NOTE: It is also possible to use PAT with a pool of addresses. In that case, instead of overloading one public address, you are overloading a small pool of public addresses.

Static NAT

Static NAT provides a one-to-one mapping between an inside address and an outside address. Static NAT allows external devices to initiate connections to internal devices. For instance, you might want to map an inside global address to a specific inside local address that is assigned to your web server, as illustrated in Figure 21-4, where Host A is communicating with Server B.

Figure 21-4 Static NAT Example



Inside Local IPv4 Address	Inside Global IPv4 Address	Outside Global IPv4 Address
10.1.1.101	209.165.201.5	209.165.202.131

Configuring static NAT translations is a simple task. You need to define the addresses to translate and then configure NAT on the appropriate interfaces. Packets that arrive on an inside interface from the identified IP address are subject to translation. Packets that arrive on an outside interface that are addressed to the identified IP address are also subject to translation.

The figure illustrates a router that is translating a source address inside a network into a source address outside the network. The numerals in Figure 21-4 correspond to the following steps that occur in translating an inside source address:

1. The user at Host A on the Internet opens a connection to Server B in the inside network. It uses Server B's public, inside global IP address, 209.165.201.5.

2. When the router receives the packet on its NAT outside-enabled interface with the inside global IPv4 address 209.165.201.5 as the destination, the router performs a NAT table lookup, using the inside global address as a key. The router then translates the address to the inside local address of Host 10.1.1.101 and forwards the packet to Host 10.1.1.101.
3. Server B receives the packet and continues the conversation.
4. The response packet that the router receives on its NAT inside-enabled interface from Server B with the source address 10.1.1.101 causes the router to check its NAT table.
5. The router replaces the inside local source address of Server B (10.1.1.101) with the translated inside global address (209.165.201.5) and forwards the packet.

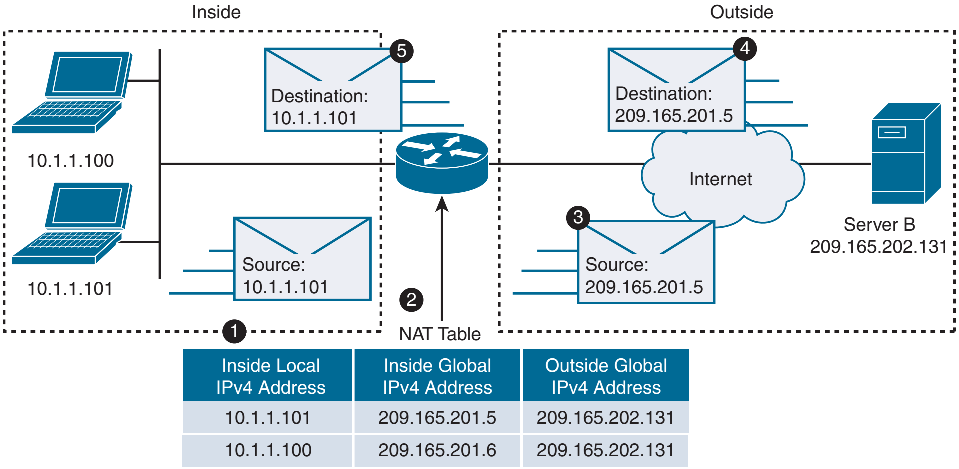
Host A receives the packet and continues the conversation. The router performs steps 2 through 5 for each packet.

Dynamic NAT

Whereas static NAT provides a permanent mapping between an internal address and a specific public address, dynamic NAT maps a group of private IP addresses to a group of public addresses. These public IP addresses come from a NAT pool. Dynamic NAT configuration differs from static NAT configuration, but it also has some similarities. Like static NAT, it requires the configuration to identify each interface as an inside interface or an outside interface. However, rather than create a static map to a single IP address, a pool of inside global addresses is used.

Figure 21-5 shows a router that is translating a source address inside a network into a source address that is outside the network.

Figure 21-5 Dynamic NAT Example



Inside Local IPv4 Address	Inside Global IPv4 Address	Outside Global IPv4 Address
10.1.1.101	209.165.201.5	209.165.202.131
10.1.1.100	209.165.201.6	209.165.202.131

The numerals in Figure 21-5 correspond to the following steps for translating an inside source address:

1. Internal users at Hosts 10.1.1.100 and 10.1.1.101 open a connection to Server B (209.165.202.131).
2. The first packet that the router receives from Host 10.1.1.101 causes the router to check its NAT table. If no static translation entry exists, the router determines that the source address 10.1.1.101 must be translated dynamically. The router then selects an outside global address (209.165.201.5) from the dynamic address pool and creates a translation entry. This type of entry is called a *simple entry*. For the second host, 10.1.1.100, the router selects a second outside global address (209.165.201.6) from the dynamic address pool and creates a second translation entry.
3. The router replaces the inside local source address of Host 10.1.1.101 with the translated inside global address 209.165.201.5 and forwards the packet. The router also replaces the inside local source address of Host 10.1.1.100 with the translated inside global address 209.165.201.6 and forwards the packet.
4. Server B receives the packet and responds to Host 10.1.1.101, using the inside global IPv4 destination address 209.165.201.5. When Server B receives the packet from Host 10.1.1.100, it responds to the inside global IPv4 destination address 209.165.201.6.
5. When the router receives the packet with the inside global IPv4 address 209.165.201.5, the router performs a NAT table lookup using the inside global address as a key. The router then translates the address back to the inside local address of Host 10.1.1.101 and forwards the packet to Host 10.1.1.101. When the router receives the packet with the inside global IPv4 address 209.165.201.6, the router performs a NAT table lookup using the inside global address as a key. The router then translates the address back to the inside local address of Host 10.1.1.100 and forwards the packet to Host 10.1.1.100.

Hosts 10.1.1.100 and 10.1.1.101 receive the packets and continue the conversations with Server B. The router performs steps 2 through 5 for each packet.

Port Address Translation (PAT)

One of the most popular forms of NAT is PAT, which is also referred to as *NAT overload* in Cisco IOS configuration. Using PAT, several inside local addresses can be translated into just one or a few inside global addresses. Most home routers operate in this manner. Your ISP assigns one address to your home router, yet several members of your family can simultaneously surf the Internet.

With PAT, multiple addresses can be mapped to one or a few addresses because a TCP or UDP port number tracks each private address. When a client opens an IP session, the NAT router assigns a port number to its source address. NAT overload ensures that clients use a different TCP or UDP port number for each client session with a server on the Internet. When a response comes back from the server, the source port number (which becomes the destination port number on the return trip) determines the client to which the router routes the packets. It also validates that the incoming packets were requested, which adds a degree of security to the session.

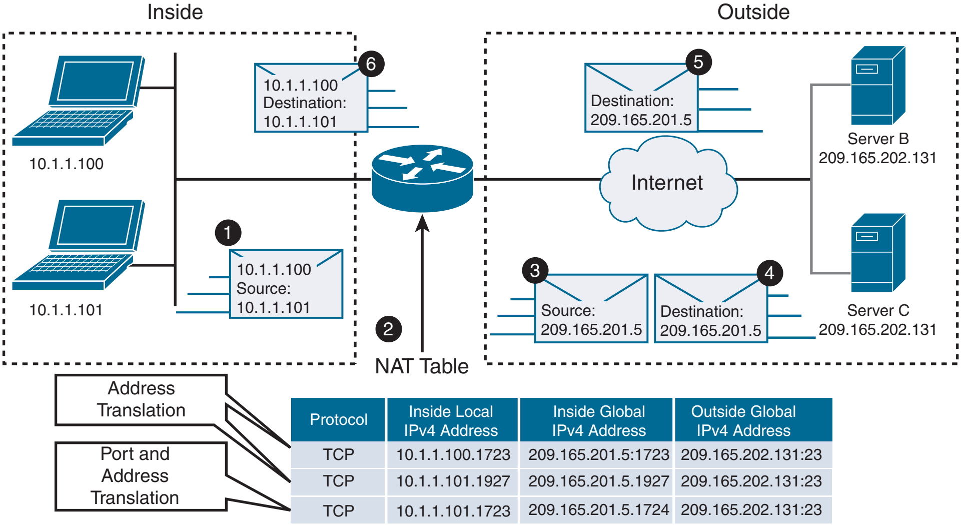
PAT has the following characteristics:

- PAT uses unique source port numbers on the inside global IPv4 address to distinguish between translations. Because the port number is encoded in 16 bits, the total number of internal addresses that NAT can translate into one external address is, theoretically, as many as 65,536.
- PAT attempts to preserve the original source port. If the source port is already allocated, PAT attempts to find the first available port number. It starts from the beginning of the appropriate port group: 0 to 511, 512 to 1023, or 1024 to 65535. If PAT does not find an available port from the appropriate port group and if more than one external IPv4 address is configured, PAT moves to the next IPv4 address and tries to allocate the original source port again. PAT continues trying to allocate the original source port until it runs out of available ports and external IPv4 addresses.

Traditional NAT routes incoming packets to their inside destination by referring to the incoming destination IP address that is given by the host on the public network. With NAT overload, there is generally only one publicly exposed IP address, so all incoming packets have the same destination IP address. Therefore, incoming packets from the public network are routed to their destinations on the private network by referring to a table in the NAT overload device that tracks public and private port pairs. This mechanism is called *connection tracking*.

Figure 21-6 illustrates a PAT operation when one inside global address represents multiple inside local addresses. The TCP port numbers act as differentiators. Internet hosts think that they are talking to a single host at the address 209.165.201.5. They are actually talking to different hosts, and the port number is the differentiator.

Figure 21-6 Port Address Translation Example


Inside
Outside
10.1.1.100
10.1.1.101
⑥
10.1.1.100
Destination:
10.1.1.101
①
10.1.1.100
Source:
10.1.1.101
②
NAT Table
⑤
Destination:
209.165.201.5
Internet
③
Source:
209.165.201.5
④
Destination:
209.165.201.5
Server B
209.165.202.131
Server C
209.165.202.131
Address Translation
Port and Address Translation

Protocol	Inside Local IPv4 Address	Inside Global IPv4 Address	Outside Global IPv4 Address
TCP	10.1.1.100.1723	209.165.201.5:1723	209.165.202.131:23
TCP	10.1.1.101.1927	209.165.201.5.1927	209.165.202.131:23
TCP	10.1.1.101.1723	209.165.201.5.1724	209.165.202.131:23

The numerals in Figure 21-6 correspond to the following steps for overloading inside global addresses:

1. The user at Host 10.1.1.100 opens a connection to Server B. A second user at Host 10.1.1.101 opens two connections to Server B.
2. The first packet that the router receives from Host 10.1.1.100 causes the router to check its NAT table. If no translation entry exists, the router determines that address 10.1.1.100 must be translated and sets up a translation of the inside local address 10.1.1.100 into an inside global address. If overloading is enabled and another translation is active, the router reuses the inside global address from that translation and saves enough information, such as port numbers, to be able to translate back. This type of entry is called an *extended entry*. The same process occurs when the router receives packets from Host 10.1.1.101.
3. The router replaces the inside local source address 10.1.1.100 with the selected inside global address 209.165.201.5, keeping the original port number 1723, and forwards the packet. A similar process occurs when the router receives packets from Host 10.1.1.101. The first Host 10.1.1.101 connection to Server B is translated into 209.165.201.5, and the router keeps its original source port number, 1927. But because its second connection has a source port number already in use, 1723, the router translates the address to 209.165.201.5 and uses a different port number, 1724.
4. Server B responds to Host 10.1.1.100, using the inside global IPv4 address 209.165.201.5 and port number 1723. Server B responds to both Host 10.1.1.101 connections with the same inside global IPv4 address as it did for Host 10.1.1.100 (209.165.201.5) and port numbers 1927 and 1724.
5. When the router receives a packet with the inside global IPv4 address 209.165.201.5, the router performs a NAT table lookup. Using the inside global address and port and outside global address and port as a key, the router translates the address back into the correct inside local address, 10.1.1.100. The router uses the same process for returning traffic destined for 10.1.1.101. Although the destination address on the return traffic is the same as it was for 10.1.1.100, the router uses the port number to determine which internal host the packet is destined for.
6. Both hosts, 10.1.1.100 and 10.1.1.101, receive their responses from Server B and continue the conversations. The router performs Steps 2 through 5 for each packet.

NAT Virtual Interface

As of Cisco IOS Software Version 12.3(14)T, Cisco introduced a new feature called NAT Virtual Interface (NVI). NVI removes the requirement to configure an interface as either inside or outside. Also, the NAT order of operations is slightly different with NVI. Classic NAT first performs routing and then translates the addresses when going from an inside interface to an outside interface and vice versa when traffic flow is reversed. NVI, however, performs routing, translation, and then routing again. NVI performs the routing operation twice—before and after translation—before forwarding the packet to an exit interface, and the whole process is symmetrical. Because of the added routing step, packets can flow from an inside interface to an inside interface (in classic NAT terms), but this process would fail with classic NAT.

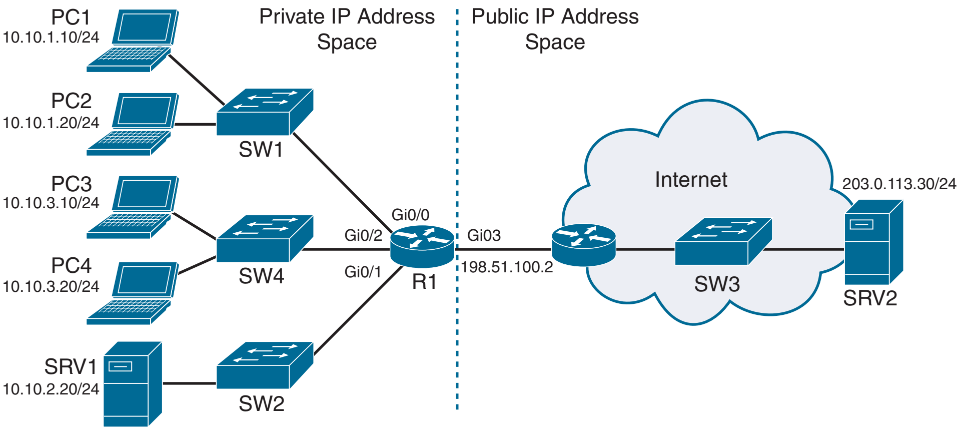
To configure interfaces to use NVI, use the **ip nat enable** interface configuration command on the inside and outside interfaces that need to perform NAT. All other NVI commands are similar to the traditional NAT commands, except for the omission of the **inside** or **outside** keywords.

NOTE: NVI is not supported on Cisco IOS XE.

NAT Configuration Example

Figure 21-7 shows the topology used for the NAT example that follows. R1 performs translation, with GigabitEthernet 0/3 as the outside interface, and GigabitEthernet 0/0, 0/1, and 0/2 as the inside interfaces.

Figure 21-7 NAT Configuration Example



Examples 21-1, 21-2, and 21-3 show the commands required to configure and verify the following deployments of NAT:

- **Static NAT on R1 so that the internal server, SRV1, can be accessed from the public Internet:** Configuring static NAT is a simple process: You define inside and outside interfaces using the **ip nat inside** and **ip nat outside** interface configuration commands and then specify which inside address should be translated to which outside address by using the **ip nat inside source static** *inside-local-address outside-global-address* global configuration command.
- **Dynamic NAT on R1 so that the internal hosts PC1 and PC2 can access the Internet by being translated into one of many possible public IP addresses:** Dynamic NAT configuration differs from static NAT configuration, but it also has some similarities. Like static NAT, dynamic NAT requires the configuration to identify each interface as an inside interface or as an outside interface. However, rather than create a static map to a single IP address, a pool of inside global addresses is used, along with an ACL that identifies which inside local addresses are to be translated. The NAT pool is defined using the command **ip nat pool** *nat-pool-name starting-ip ending-ip* {**netmask** *netmask* | **prefix-length** *prefix-length*}. If the router needs to advertise the pool in a dynamic routing protocol, you can add the add-route argument at the end of the **ip nat pool** command to add a static route in the router's routing table for the pool that can be redistributed into the dynamic routing protocol.

  The ACL-to-NAT pool mapping is defined by the **ip nat inside source list** *acl* **pool** *nat-pool-name* global configuration command. Instead of using an ACL, it is possible to match traffic based on route map criteria by using the **ip nat inside source route-map** command.

- **Port Address Translation on R1 so that the internal hosts PC3 and PC4 can access the Internet by sharing a single public IP address:** To configure PAT, identify inside and outside interfaces by using the **ip nat inside** and **ip nat outside** interface configuration commands, respectively. An ACL must be configured to match all inside local addresses that need to be translated, and NAT needs to be configured so that all inside local addresses are translated to the address of the outside interface. This is achieved by using the **ip nat inside source list** *acl* {**interface** *interface-id* | **pool** *nat-pool-name*} **overload** global configuration command.

Example 21-1 Configuring Static NAT

```
R1(config)# interface GigabitEthernet 0/1
R1(config-if)# ip nat inside
R1(config-if)# interface GigabitEthernet 0/3
R1(config-if)# ip nat outside
R1(config-if)# exit
R1(config)# ip nat inside source static 10.10.2.20 198.51.100.20
R1(config)# end
```

```
SRV2# telnet 198.51.100.20
Trying 198.51.100.20 ... Open

User Access Verification

Username: admin
Password: Cisco123
SRV1>
```

```
R1# show ip nat translations
Pro  Inside global       Inside local       Outside local        Outside global
tcp  198.51.100.20:23    10.10.2.20:23      203.0.113.30:23024   203.0.113.30:23024
---  198.51.100.20       10.10.2.20         ---                  ---
```

Example 21-1 shows a Telnet session established between SRV2 and SRV1 after the static NAT entry is configured. The **show ip nat translations** command displays two entries in the router's NAT table. The first entry is an extended entry because it includes more details than just a public IP address that is mapping to a private IP address. In this case, it specifies the protocol (TCP) and also the ports in use on both systems. The extended entry is due to the use of the static translation for the Telnet session from SRV1 to SRV2. It details the characteristics of that session.

The second entry is a simple entry that maps one IP address to another. The simple entry is the persistent entry that is associated with the configured static translation.

Example 21-2 Configuring Dynamic NAT

```
R1(config)# access-list 10 permit 10.10.1.0 0.0.0.255
R1(config)# interface GigabitEthernet 0/0
R1(config-if)# ip nat inside
R1(config-if)# exit
R1(config)# ip nat pool NatPool 198.51.100.100 198.51.100.149 netmask
  255.255.255.0
R1(config)# ip nat inside source list 10 pool NatPool
R1(config)# end
```

```
PC1# ping 203.0.113.30
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 203.0.113.30, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms


R1# show ip nat translations
Pro  Inside global        Inside local       Outside local        Outside global
icmp 198.51.100.100:4     10.10.1.10:4       203.0.113.30:4       203.0.113.30:4
---  198.51.100.100       10.10.1.10         ---                  ---
---  198.51.100.20        10.10.2.20         ---                  ---
```

Example 21-2 shows an ICMP ping sent from PC1 to SRV2.

There are now three translations in R1's NAT table:

- The first is an extended translation that is associated with the ICMP session. This entry is usually short-lived and can time out quickly compared to a TCP entry.
- The second is a simple entry in the table that is associated with the assignment of an address from the pool to PC1.
- The third entry that is translating 10.10.2.20 to 198.51.100.20 is the static entry from Example 21-1.

Example 21-3 Configuring PAT

```
R1(config)# access-list 20 permit 10.10.3.0 0.0.0.255
R1(config)# interface GigabitEthernet 0/2
R1(config-if)# ip nat inside
R1(config-if)# exit
R1(config)# ip nat inside source list 20 interface gigabitethernet0/2 overload
```

```
PC3# telnet 203.0.113.30
Trying 203.0.113.30 ... Open


User Access Verification

Username: admin
Password: Cisco123
SRV2>
```

```
PC4# telnet 203.0.113.30
Trying 203.0.113.30 ... Open


User Access Verification

Username: admin
Password: Cisco123
SRV2>
```

```
R1# show ip nat translations
Pro  Inside global        Inside local         Outside local        Outside global
---  198.51.100.20        10.10.2.20           ---                  ---
tcp  198.51.100.2:21299   10.10.3.10:21299     203.0.113.30:23      203.0.113.30:23
tcp  198.51.100.2:34023   10.10.3.20:34023     203.0.113.30:23      203.0.113.30:23
```

In Example 21-3, R1 is using the inside TCP source port to uniquely identify the two translation sessions: one from PC3 to SRV2 using Telnet and one from PC4 to SRV2 using Telnet.

When R1 receives a packet from SRV2 (203.0.113.30) with source port 23 that is destined for 198.51.100.2 and destination port 21299, R1 knows to translate the destination address to 10.10.3.10 and forward the packet to PC3. On the other hand, if the destination port of a similar inbound packet is 34023, R1 translates the destination address to 10.10.3.20 and forwards the packet to PC4.

Tuning NAT

A router keeps NAT entries in the translation table for a configurable length of time. For TCP connections, the default timeout period is 86,400 seconds, or 24 hours. Because UDP is not connection based, the default timeout period is much shorter: only 300 seconds, or 5 minutes. The router removes translation table entries for DNS queries after only 60 seconds.

You can adjust these parameters by using the **ip nat translation** command, which accepts arguments in seconds, as shown in Example 21-4:

Example 21-4 Tuning NAT

```
Router(config)# ip nat translation ?
  dns-timeout              Specify timeout for NAT DNS flows
  finrst-timeout           Specify timeout for NAT TCP flows after a FIN or RST
  icmp-timeout             Specify timeout for NAT ICMP flows
  max-entries              Specify maximum number of NAT entries
  tcp-timeout              Specify timeout for NAT TCP flows
  timeout                  Specify timeout for dynamic NAT translations
  udp-timeout              Specify timeout for NAT UDP flows
```

To remove dynamic entries from the NAT translation table, use the **clear ip nat translation** command. You can also use the **debug ip nat** command to monitor the NAT process for any errors.

Network Time Protocol

NTP is used to synchronize timekeeping among a set of distributed time servers and clients. NTP uses UDP port 123 as both the source and destination ports. NTP runs over IPv4 but, in the case of NTPv4, it can also run over IPv6.

An NTP network usually gets its time from an authoritative time source, such as a radio clock or an atomic clock that is attached to a time server. NTP then distributes this time across the network. An NTP client makes a transaction with its server over its polling interval (from 64 to 1024 seconds), which dynamically changes over time, depending on the network conditions between the NTP server and the client. No more than one NTP transaction per minute is needed to synchronize two machines.

The communications between machines running NTP (associations) are usually statically configured. Each machine is given the IP addresses of all machines with which it should form associations. However, in a LAN, NTP can be configured to use IP broadcast messages instead. This alternative reduces configuration complexity because each machine can be configured to send or receive broadcast messages. However, the accuracy of timekeeping is marginally reduced because the information flows one way only.

NTP Versions

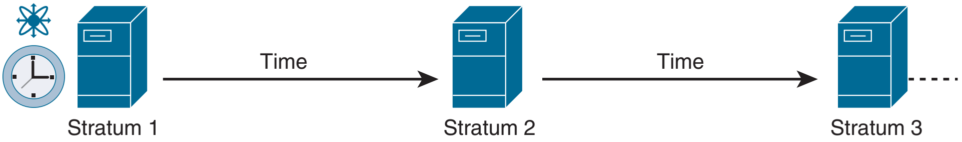
NTPv4 is an extension of NTPv3 and provides the following capabilities:

- NTPv4 supports IPv6, making NTP time synchronization possible over IPv6.
- Security is improved over NTPv3. NTPv4 provides a whole security framework that is based on public key cryptography and standard X.509 certificates.

- Using specific multicast groups, NTPv4 can automatically calculate its time-distribution hierarchy through an entire network. NTPv4 automatically configures the hierarchy of the servers to achieve the best time accuracy for the lowest bandwidth cost.
- In NTPv4 for IPv6, IPv6 multicast messages instead of IPv4 broadcast messages are used to send and receive clock updates.

NTP uses the concept of a *stratum* to describe how many NTP hops away a machine is from an authoritative time source. For example, a stratum 1 time server has a radio or atomic clock that is directly attached to it. It sends its time to a stratum 2 time server through NTP, as illustrated in Figure 21-8. A machine running NTP automatically chooses the machine with the lowest stratum number that it is configured to communicate with, using NTP as its time source. This strategy effectively builds a self-organizing tree of NTP speakers.

Figure 21-8 NTP Stratum Example



NTP performs well over the nondeterministic path lengths of packet-switched networks because it makes robust estimates of the following three key variables in the relationship between a client and a time server:

- **Network delay:** The amount of time it takes for traffic to flow between the client and the time server.
- **Dispersion of time packet exchanges:** A measure of maximum clock error between the two hosts.
- **Clock offset:** The correction that is applied to a client clock to synchronize it.

Clock synchronization at the 10-millisecond level over long-distance WANs (124.27 miles [200 km]) and at the 1-millisecond level for LANs is routinely achieved.

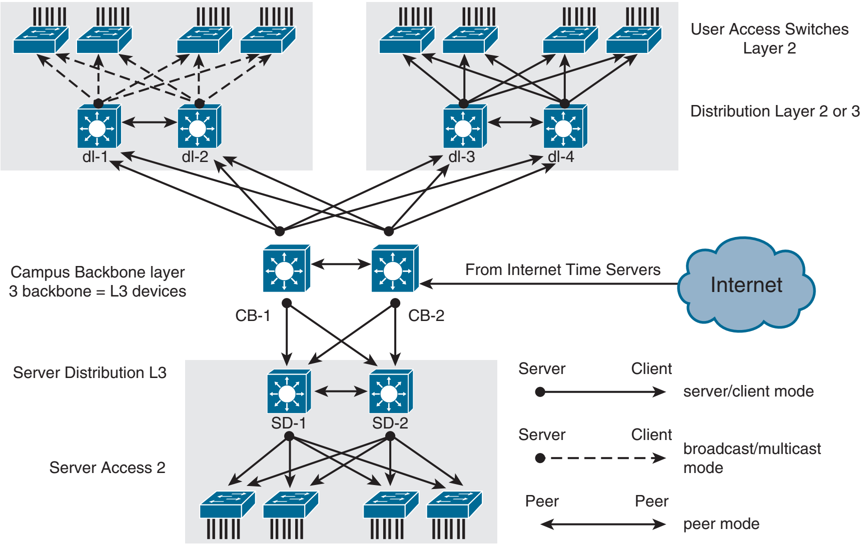
NTP avoids synchronizing to a machine whose time may not be accurate in two ways:

- NTP never synchronizes to a machine that is not itself synchronized.
- NTP compares the time that is reported by several machines, and it does not synchronize to a machine whose time is significantly different from the others, even if its stratum is lower.

NTP Modes

NTP can operate in four different modes that provide flexibility for configuring time synchronization in a network. Figure 21-9 shows these four modes deployed in an enterprise network.

Figure 21-9 NTP Modes


User Access Switches
Layer 2
Distribution Layer 2 or 3
dl-1
dl-2
dl-3
dl-4
Campus Backbone layer
3 backbone = L3 devices
From Internet Time Servers
Internet
CB-1
CB-2
Server Distribution L3
SD-1
SD-2
Server Access 2
Server
Client
server/client mode
Server
Client
broadcast/multicast
mode
Peer
Peer
peer mode

NTP Server

An NTP server provides accurate time information to clients. If using a Cisco device as an authoritative clock, use the **ntp master** command.

NTP Client

An NTP client synchronizes its time to the NTP server. The client mode is most suited for servers and clients that are not required to provide any form of time synchronization to other local clients. Clients can also be configured to provide accurate time to other devices.

The server and client modes are usually combined to operate together. A device that is an NTP client can act as an NTP server to another device. The client/server mode is a common network configuration. A client sends a request to the server and expects a reply at some future time. This process could also be called a *poll operation* because the client polls the time and authentication data from the server.

You configure a client in client mode by using the **ntp server** command and specifying the DNS name or address of the server. The server requires no prior configuration. In a common client/server model, a client sends an NTP message to one or more servers and processes the replies as received. The server exchanges addresses and ports, overwrites certain fields in the message, recalculates the checksum, and returns the message immediately. The information that is included in the NTP message allows the client to determine the server time in comparison to local time and adjust the local clock accordingly. In addition, the message includes information to calculate the expected timekeeping accuracy and reliability and to select the best server.

NTP Peer

Peers exchange time synchronization information. The peer mode, which is commonly known as *symmetric mode*, is intended for configurations where a group of low-stratum peers operate as mutual backups for each other.

Each peer operates with one or more primary reference sources, such as a radio clock or a subset of reliable secondary servers. If one of the peers loses all the reference sources or simply ceases to operate, the other peers automatically reconfigure so that time values can flow from the surviving peers to all the others in the group. In some contexts, this operation is described as *push/pull*, in that the peer either pulls or pushes the time and values, depending on the particular configuration.

Symmetric modes are most often used between two or more servers operating as a mutually redundant group and are configured with the **ntp peer** command. In these modes, the servers in the group arrange the synchronization paths for maximum performance, depending on network jitter and propagation delay. If one or more of the group members fail, the remaining members automatically reconfigure as required.

Broadcast/Multicast

Broadcast/multicast mode is a special push mode for the NTP server. Where the requirements in accuracy and reliability are modest, clients can be configured to use broadcast or multicast modes. Normally, these modes are not utilized by servers with dependent clients. The advantage is that clients do not need to be configured for a specific server, so all operating clients can use the same configuration file.

Broadcast mode requires a broadcast server on the same subnet. Because broadcast messages are not propagated by routers, only broadcast servers on the same subnet are used. Broadcast mode is intended for configurations that involve one or a few servers and a potentially large client population. On a Cisco device, a broadcast server is configured by using the **ntp broadcast** command with a local subnet address. A Cisco device acting as a broadcast client is configured by using the **ntp broadcast client** command, allowing the device to respond to broadcast messages received on any interface.

Figure 21-9 shows a high-stratum campus network taken from the standard Cisco campus network design that contains three components. The campus core consists of two Layer 3 devices labeled CB-1 and CB-2. The data center component, located in the lower section of the figure, has two Layer 3 routers, labeled SD-1 and SD-2. The remaining devices in the server block are Layer 2 devices. In the upper left, there is a standard access block with two Layer 3 distribution devices labeled dl-1 and dl-2. The remaining devices are Layer 2 switches. In this client access block, the time is distributed using the broadcast option. In the upper right is another standard access block that uses a client/server time distribution configuration.

The campus backbone devices are synchronized to the Internet time servers in a client/server model.

Notice that all distribution layer switches are configured in a client/server relationship with the Layer 3 core switches, but the distribution switches are also peering with each other; the same applies to the two Layer 3 core switches. This offers an extra level of resilience.

NTP Source Address

The source of an NTP packet is the same as the interface that the packet was sent out on. When you implement authentication and access lists, it is good to have a specific interface set to act as the source interface for NTP.

It would be wise to choose a loopback interface to use as the NTP source. The loopback interface will never be down, as physical interfaces may be.

If you configure Loopback 0 to act as the NTP source for all communication, and that interface has, for example, IP address 192.168.12.31, you can write up just one access list to allow or deny based on the single IP address 192.168.12.31.

Use the **ntp source** global configuration command to specify which interface to use as the source IP address for NTP packets.

Securing NTP

NTP can be an easy target in a network. Because device certificates rely on accurate time, you should secure NTP operation. You can secure NTP operation by using authentication and access lists.

NTP Authentication

Cisco devices support only MD5 authentication for NTP. To configure NTP authentication, follow these steps:

1. Define the NTP authentication key or keys with the **ntp authentication-key** *key-id* **md5** *key-string* command. Every number specifies a unique NTP key.
2. Enable NTP authentication by using the **ntp authenticate** command.
3. Tell the device which keys are valid for NTP authentication by using the **ntp trusted-key** *key-id* command. The only argument to this command is the key defined in step 1.
4. Specify the NTP server that requires authentication by using the **ntp server** *server-ip-address* **key** *key-id* command. You can similarly authenticate NTP peers by using this command.

Not all clients need to be configured with NTP authentication. NTP does not authenticate clients; it authenticates the source. Because of this, the device still responds to unauthenticated requests, so it is important to use access lists to limit NTP access.

After implementing authentication for NTP, use the **show ntp status** command to verify that the clock is still synchronized. If a client has not successfully authenticated the NTP source, the clock will be unsynchronized.

NTP Access Lists

Once a router or switch is synchronized to NTP, the source acts as an NTP server to any device that requests synchronization. You should configure access lists on devices that synchronize their time with external servers. Why would you want to do that? A lot of NTP synchronization requests from the Internet might overwhelm your NTP server device. An attacker could use NTP queries to discover the time servers to which your device is synchronized and then, through an attack such as DNS cache poisoning, redirect your device to a system under its control. If an attacker modifies time on your devices, that can confuse any time-based security implementations you might have in place.

For NTP, the following four restrictions can be configured through access lists when using the **ntp access-group** global configuration command:

- **peer:** Time synchronization requests and control queries are allowed. A device is allowed to synchronize itself to remote systems that pass the access list.
- **serve:** Time synchronization requests and control queries are allowed. A device is not allowed to synchronize itself to remote systems that pass the access list.
- **serve-only:** It allows synchronization requests only.
- **query-only:** It allows control queries only.

Say that you have a hierarchical model with two routers configured to provide NTP services to the rest of the devices in your network. You would configure these two routers with **peer** and **serve-only** restrictions. You would use the **peer** restriction mutually on the two core routers. You would use the **serve-only** restriction on both core routers to specify which devices in the network are allowed to synchronize their information with these two routers.

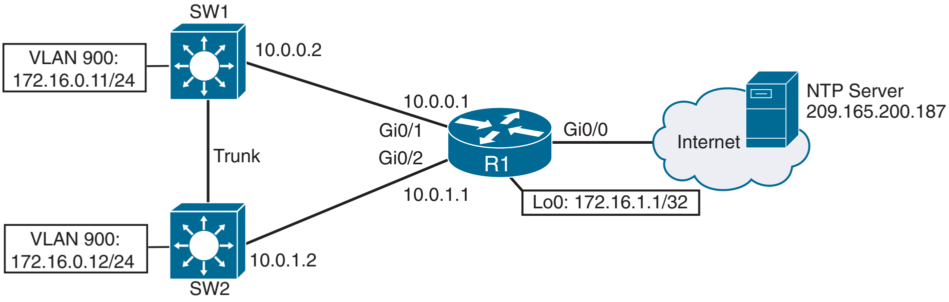
If your device is configured as the NTP master, then you must allow access to the source IP address 127.127.x.1 because 127.127.x.1 is the internal server that is created by the **ntp master** command. (The value of the third octet varies between platforms.)

After you secure the NTP server with access lists, make sure to check whether the clients still have their clocks synchronized via NTP by using the **show ntp status** command. You can verify which IP address was assigned to the internal server by using the **show ntp associations** command.

NTP Configuration Example

Figure 21-10 shows the topology used for the NTP configuration example that follows.

Figure 21-10 NTP Configuration Example Topology



Example 21-5 shows the commands used to deploy NTP. In this example, R1 synchronizes its time with the NTP server. SW1 and SW2 synchronize their time with R1, but SW1 and SW2 also peer with each other for further NTP resiliency. The NTP source interface option is used to allow for predictability when configuring the NTP ACL.

Example 21-5 Configuring NTP

```
R1(config)# ntp source Loopback 0
R1(config)# ntp server 209.165.200.187
R1(config)# access-list 10 permit 209.165.200.187
R1(config)# access-list 10 permit 172.16.0.11
R1(config)# access-list 10 permit 172.16.0.12
R1(config)# ntp access-group peer 10
```

```
SW1(config)# ntp source Vlan 900
SW1(config)# ntp server 172.16.1.1
SW1(config)# ntp peer 172.16.0.12
SW1(config)# access-list 10 permit 172.16.1.1
SW1(config)# access-list 10 permit 172.16.0.12
SW1(config)# ntp access-group peer 10
```

```
SW2(config)# ntp source Vlan 900
SW2(config)# ntp server 172.16.1.1
SW2(config)# ntp peer 172.16.0.11
SW2(config)# access-list 10 permit 172.16.1.1
SW2(config)# access-list 10 permit 172.16.0.11
SW2(config)# ntp access-group peer 10
```

Example 21-6 displays the output from the **show ntp status** command issued on R1, SW1, and SW2.

Example 21-6 Verifying NTP Status

```
R1# show ntp status
Clock is synchronized, stratum 2, reference is 209.165.200.187
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is 2**10
ntp uptime is 1500 (1/100 of seconds), resolution is 4000
reference time is D67E670B.0B020C68 (05:22:19.043 PST Mon Jan 13 2014)
clock offset is 0.0000 msec, root delay is 0.00 msec
root dispersion is 630.22 msec, peer dispersion is 189.47 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000592842 s/s
system poll interval is 64, last update was 5 sec ago.
```

```
SW1# show ntp status
Clock is synchronized, stratum 3, reference is 172.16.1.1
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is 2**18
ntp uptime is 1500 (1/100 of seconds), resolution is 4000
```

```
reference time is D67FD8F2.4624853F (10:40:34.273 EDT Tue Jan 14 2014)
clock offset is 0.0053 msec, root delay is 0.00 msec
root dispersion is 17.11 msec, peer dispersion is 0.02 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000049563 s/s
system poll interval is 64, last update was 12 sec ago.
```

```
SW2# show ntp status
Clock is synchronized, stratum 3, reference is 172.16.1.1
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is 2**18
ntp uptime is 1500 (1/100 of seconds), resolution is 4000
reference time is D67FD974.17CE137F (10:42:44.092 EDT Tue Jan 14 2014)
clock offset is 0.0118 msec, root delay is 0.00 msec
root dispersion is 17.65 msec, peer dispersion is 0.02 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000003582 s/s
system poll interval is 64, last update was 16 sec ago.
```

The output in Example 21-5 shows that NTP has successfully synchronized the clock on the devices. The stratum will be +1 in comparison to the NTP source. Because the output for R1 shows that this device is stratum 2, you can assume that R1 is synchronizing to a stratum 1 device.

Example 21-7 displays the output from the **show ntp associations** command issued on R1, SW1, and SW2.

Example 21-7 Verifying NTP Associations

```
 R1# show ntp associations

  address         ref clock        st   when   poll reach  delay  offset   disp
*~209.165.200.187 .LOCL.            1     24     64    17  1.000  -0.500  2.820
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~ configured
```

```
SW1# show ntp association

  address         ref clock         st   when  poll reach   delay  offset    disp
*~10.0.0.1        209.165.200.187    2     22   128   377     0.0    0.02     0.0
+~172.16.0.12     10.0.1.1           3      1   128   376     0.0   -1.00     0.0
 * master (synced), # master (unsynced), + selected, - candidate, ~ configured
```

```
SW2# show ntp association

  address          ref clock         st   when   poll  reach   delay   offset     disp
*~10.0.1.1         209.165.200.187    2     18    128    377     0.0     0.02      0.3
+~172.16.0.11      10.0.0.1           3      0    128     17     0.0    -3.00   1875.0
 * master (synced), # master (unsynced), + selected, - candidate, ~ configured
```

The output in Example 21-6 shows each device's NTP associations. The ***** before the IP address signifies that the devices are associated with that server. If you have multiple NTP servers defined, others will be marked with **+**, which signifies alternate options. Alternate servers are the servers that become associated if the currently associated NTP server fails. In this case, SW1 and SW2 are peering with each other, as well as with R1.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	15
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	8, 11
Network Time Protocol: Best Practices White Paper	https://www.cisco.com/c/en/us/support/docs/availability/high-availability/19643-ntpm.html

GRE and IPsec

ENCOR 350-401 Exam Topics

Virtualization

- Configure and verify data path virtualization technologies
  - GRE and IPsec tunneling

Key Topics

Today we review two overlay network technologies: Generic Routing Encapsulation (GRE) and Internet Protocol Security (IPsec). An *overlay* network is a virtual network that is built on top of an *underlay* network. The underlay is a traditional network, which provides connectivity between network devices such as routers and switches. In the case of GRE and IPsec, the overlay is most often represented as tunnels or virtual private networks (VPNs) that are built on top of a public insecure network such as the Internet. These tunnels overcome segmentation and security shortcomings of traditional networks.

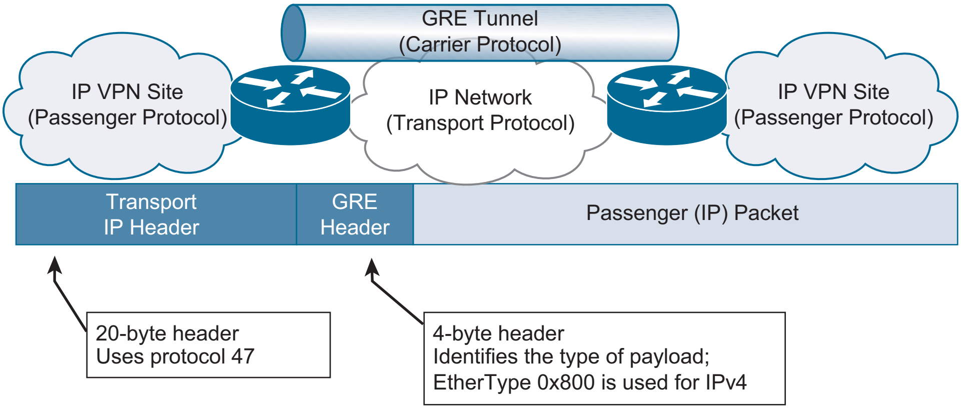
Generic Routing Encapsulation

GRE is a tunneling protocol that provides a secure path for transporting packets over a public network by encapsulating packets inside a transport protocol. GRE supports multiple Layer 3 protocols, such as IP, IPX, and AppleTalk. It also enables the use of multicast routing protocols across the tunnel.

GRE adds a 20-byte transport IP header and a 4-byte GRE header and hides the existing packet headers, as illustrated in Figure 20-1. The GRE header contains a flag field and a protocol type field to identify the Layer 3 protocol being transported. It may contain a tunnel checksum, tunnel key, and tunnel sequence number.

GRE does not encrypt traffic or use any strong security measures to protect the traffic. GRE supports both IPv4 and IPv6 addresses as either the underlay or overlay network. In Figure 20-1, the IP network cloud is the underlay, and the GRE tunnel is the overlay. The passenger protocol is the carrier between VPN sites, moving, for example, user data and routing protocol updates. Because of the added encapsulation overhead when using GRE, you might have to adjust the MTU (Maximum Transmission Unit) setting on GRE tunnels by using the **ip mtu** interface configuration command. The MTU setting must match on both sides.

Figure 20-1 GRE Encapsulation


GRE Tunnel
(Carrier Protocol)
IP VPN Site
(Passenger Protocol)
IP Network
(Transport Protocol)
IP VPN Site
(Passenger Protocol)
Transport
IP Header
GRE
Header
Passenger (IP) Packet
20-byte header
Uses protocol 47
4-byte header
Identifies the type of payload;
EtherType 0x800 is used for IPv4

Generally, a tunnel is a logical interface that provides a way to encapsulate passenger packets inside a transport protocol. A GRE tunnel is a point-to-point tunnel that allows a wide variety of passenger protocols to be transported over the IP network. GRE tunnels enable you to connect branch offices across the Internet or a wide-area network (WAN). The main benefit of a GRE tunnel is that it supports IP multicast and therefore is appropriate for tunneling routing protocols.

GRE can be used along with IPsec to provide authentication, confidentiality, and data integrity. GRE over IPsec tunnels is typically configured in a hub-and-spoke topology over an untrusted WAN in order to minimize the number of tunnels that each router must maintain.

GRE, originally developed by Cisco, is designed to encapsulate arbitrary types of network layer packets inside arbitrary types of network layer packets, as defined in RFC 1701, "Generic Routing Encapsulation (GRE)"; RFC 1702, "Generic Routing Encapsulation over IPv4 Networks"; and RFC 2784, "Generic Routing Encapsulation (GRE)."

GRE Configuration Steps

To implement a GRE tunnel, you perform the following actions:

1. Create a tunnel interface by using the following commands:

```
Router(config)# interface tunnel tunnel-id
```

2. Configure GRE tunnel mode. GRE IPv4 is the default tunnel mode, so it is not necessary to configure it. Other options include GRE IPv6:

```
Router(config-if)# tunnel mode gre ip
```

3. Configure an IP address for the tunnel interface by using the following command:

```
Router(config-if)# ip address ip-address mask
```

This address is part of the overlay network.

4. Use the following command to specify the tunnel source IP address, which is the address assigned to the local interface in the underlay network (and can be a physical or loopback interface, as long as it is reachable from the remote router):

   ```
   Router(config-if)# tunnel source {ip-address | interface-id}
   ```

5. Use the following command to specify the tunnel destination IP address, which is the address that is assigned to the remote router in the underlay network:

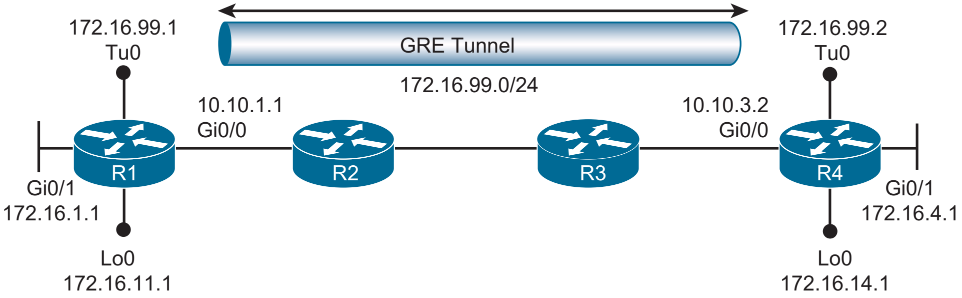
   ```
   Router(config-if)# tunnel destination ip-address
   ```

The minimum GRE tunnel configuration requires specification of the tunnel source address and destination address. Optionally, you can specify the bandwidth and keepalive values, and you can lower the IP MTU setting. The default bandwidth of a tunnel interface is 100 Kbps, and the default keepalive is every 10 seconds, with three retries. A typical value used for the MTU on a GRE interface is 1400 bytes.

GRE Configuration Example

Figure 20-2 shows the topology used for the configuration example that follows. A GRE tunnel using 172.16.99.0/24 is established between R1 and R4 across the underlay network through R2 and R3. Once the tunnel is configured, OSPF is enabled on R1 and R4 to advertise their respective Loopback 0 and GigabitEthernet 0/1 networks.

Figure 20-2 GRE Configuration Example Topology



Example 20-1 shows the commands required to configure a GRE tunnel between R1 and R4.

Example 20-1 Configuring GRE on R1 and R4

```
R1(config)# interface Tunnel 0
R1(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed state to down
R1(config-if)# ip address 172.16.99.1 255.255.255.0
R1(config-if)# tunnel source 10.10.1.1
R1(config-if)# tunnel destination 10.10.3.2
R1(config-if)# ip mtu 1400
```

```
R1(config-if)# bandwidth 1000
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed state to up
R1(config-if)# exit
R1(config)# router ospf 1
R1(config-router)# router-id 0.0.0.1
R1(config-router)# network 172.16.99.0 0.0.0.255 area 0
R1(config-router)# network 172.16.1.0 0.0.0.255 area 1
R1(config-router)# network 172.16.11.0 0.0.0.255 area 1
```

```
R4(config)# interface Tunnel 0
R4(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed state to down
R4(config-if)# ip address 172.16.99.2 255.255.255.0
R4(config-if)# tunnel source GigabitEthernet 0/0
R4(config-if)# tunnel destination 10.10.1.1
R4(config-if)# ip mtu 1400
R4(config-if)# bandwidth 1000
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed state to up
R4(config-if)# exit
R4(config)# router ospf 1
R4(config-router)# router-id 0.0.0.1
R4(config-router)# network 172.16.99.0 0.0.0.255 area 0
R4(config-router)# network 172.16.4.0 0.0.0.255 area 4
R4(config-router)# network 172.16.14.0 0.0.0.255 area 4
```

In Example 20-1, each router is configured with a tunnel interface in the 172.16.99.0/24 subnet. The tunnel source and destination are also configured, but notice that on R4, the interface—instead of the IP address—is used as the tunnel source. This demonstrates both configuration options for the tunnel source. Both routers are also configured with a lower MTU of 1400 bytes, and the bandwidth has been increased to 1000 Kbps, or 1 Mbps. Finally, OSPF is configured with Area 0 used across the GRE tunnel, Area 1 is used on R1's LANs, and Area 4 is used on R4's LANs.

To determine whether the tunnel interface is up or down, use the **show ip interface brief** command.

You can verify the state of a GRE tunnel by using the **show interface tunnel** command. The line protocol on a GRE tunnel interface is up as long as there is a route to the tunnel destination.

By issuing the **show ip route** command, you can identify the route between the GRE tunnel-enabled routers. Because a tunnel is established between the two routers, the path is seen as being directly connected.

Example 20-2 shows these verification commands applied to the GRE configuration example.

Example 20-2 Verifying GRE on R1 and R4

```
R1# show ip interface brief Tunnel 0
Interface                  IP-Address      OK? Method Status           Protocol
Tunnel0                    172.16.99.1     YES manual up               up
```

```
R4# show interface Tunnel 0
Tunnel0 is up, line protocol is up
  Hardware is Tunnel
  Internet address is 172.16.99.2/24
  MTU 17916 bytes, BW 1000 Kbit/sec, DLY 50000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel source 10.10.3.2 (GigabitEthernet0/0), destination 10.10.1.1
  Tunnel protocol/transport GRE/IP
<... output omitted ...>
```

```
R1# show ip route

<... output omitted ...>
C        10.10.1.0/24 is directly connected, GigabitEthernet0/0
L        10.10.1.1/32 is directly connected, GigabitEthernet0/0
      172.16.0.0/16 is variably subnetted, 8 subnets, 2 masks
C        172.16.1.0/24 is directly connected, GigabitEthernet0/1
L        172.16.1.1/32 is directly connected, GigabitEthernet0/1
O        172.16.4.0/24 [110/101] via 172.16.99.2, 00:19:23, Tunnel0
C        172.16.11.0/24 is directly connected, Loopback0
L        172.16.11.1/32 is directly connected, Loopback0
O        172.16.14.1/32 [110/101] via 172.16.99.2, 00:19:23, Tunnel0
C        172.16.99.0/24 is directly connected, Tunnel0
L        172.16.99.1/32 is directly connected, Tunnel0
```

```
R1# show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
0.0.0.4           0   FULL/  -        00:00:37    172.16.99.2     Tunnel0
```

In the output in Example 20-2, notice that the tunnel interface is up and operating in IPv4 GRE mode. The OSPF point-to-point neighbor adjacency is established between R1 and R4 across the GRE tunnel. Because the tunnel has a bandwidth of 1000 Kbps, the total cost from R1 to reach

R4's Loopback 0 and GigabitEthernet 0/1 networks is 101—that is, 100 for the tunnel cost, and 1 for the interface costs since loopback and GigabitEthernet interfaces each have a default cost of 1.

Note that although it is not explicitly shown, this configuration example assumes that connectivity exists across the underlay network to allow R1 and R4 to reach each other's GigabitEthernet 0/0 interfaces; otherwise, the overlay GRE tunnel would fail.

IP Security (IPsec)

Enterprises use site-to-site VPNs as a replacement for a classic private WAN to either connect geographically dispersed sites of the same enterprise or to connect to their partners over a public network. This type of connection lowers costs while providing scalable performance. Site-to-site VPNs authenticate VPN peers and network devices that provide VPN functionality for an entire site and provide secure data transmission between sites over an untrusted network, such as the Internet. This section describes secure site-to-site connectivity solutions and looks at different IPsec VPN configuration options available on Cisco routers.

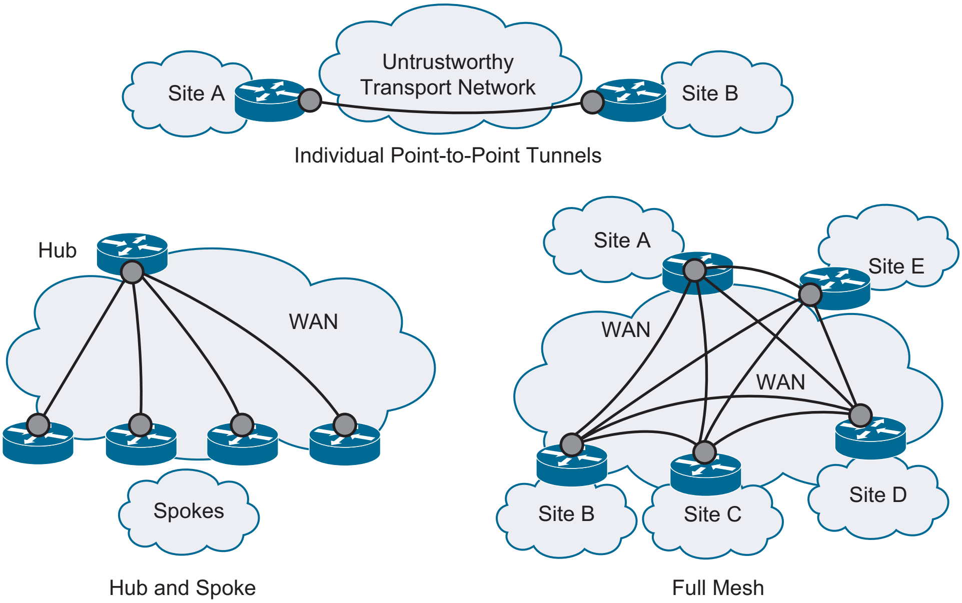
Site-to-Site VPN Technologies

VPNs allow enterprise networks to be expanded across uncontrolled network segments—typically across WAN segments.

A network is the interconnection of network nodes (typically routers). With most VPN technologies, this interconnection is largely a logical one because the physical interconnection of network devices is of no consequence to how the VPN protocols create connectivity between network users.

Figure 20-3 illustrates the three typical logical VPN topologies that are used in site-to-site VPNs:

Figure 20-3 Site-to-Site VPN Topologies



@cisco_in_persian_channel

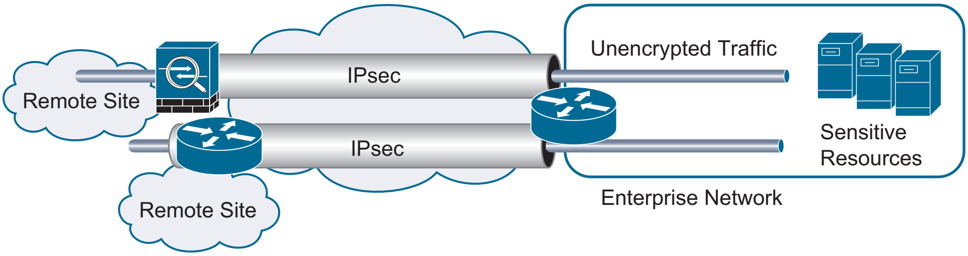
- **Individual point-to-point VPN connection:** Two sites interconnect using a secure VPN path. The network may include a few individual point-to-point VPN connections to connect sites that require mutual connectivity.
- **Hub-and-spoke network:** One central site is considered a hub, and all other sites (spokes) peer exclusively with the central site devices. Typically, most of the user traffic flows between the spoke network and the hub network, although the hub may be able to act as a relay and facilitate spoke-to-spoke communication over the hub.
- **Fully meshed network:** Every network device is connected to every other network device. This topology enables any-to-any communication; provides the most optimal, direct paths in the network; and provides the greatest flexibility to network users.

In addition to the three main VPN topologies, these other more complex topologies can be created as combination topologies:

- **Partial mesh:** A network in which some devices are organized in a full mesh topology, and other devices form either hub-and-spoke or point-to-point connections to some of the fully meshed devices. A partial mesh does not provide the level of redundancy of a full mesh topology, but it is less expensive to implement. Partial mesh topologies are generally used in peripheral networks that connect to a fully meshed backbone.
- **Tiered hub-and-spoke:** A network of hub-and-spoke topologies in which a device can behave as a hub in one or more topologies and a spoke in other topologies. Traffic is permitted from spoke groups to their most immediate hub.
- **Joined hub-and-spoke:** A combination of two topologies (hub-and-spoke, point-to-point, or full mesh) that connect to form a point-to-point tunnel. For example, a joined hub-and-spoke topology could comprise two hub-and-spoke topologies, with the hubs acting as peer devices in a point-to-point topology.

Figure 20-4 illustrates a simple enterprise site-to-site VPN scenario. An enterprise may use a site-to-site VPN as a replacement for a classic routed WAN to either connect geographically dispersed sites of the same enterprise or to connect to their partners over a public network. This type of connection lowers costs while providing scalable performance. A site-to-site VPN authenticates VPN peers and network devices that provide VPN functionality for an entire site and provides secure transmission between sites over an untrusted network such as the Internet.

Figure 20-4 Site-to-Site IPsec VPN Scenario



To control traffic that flows over site-to-site VPNs, VPN devices use basic firewall-like controls to limit connectivity and prevent traffic spoofing. These networks often work over more controlled transport networks and usually do not encounter many problems with traffic filtering in transport networks between VPN endpoints. However, because these networks provide core connectivity in an enterprise network, they often must provide high-availability and high-performance functions to critical enterprise applications.

There are several site-to-site VPN solutions, and each of them enables the site-to-site VPN to operate in a different way. For example, the Cisco DMVPN (Dynamic Multipoint VPN) solution enables site-to-site VPNs without a permanent VPN connection between sites and can dynamically create IPsec tunnels. Another solution, FlexVPN, uses the capabilities of IKEv2 (Internet Key Exchange Version 2).

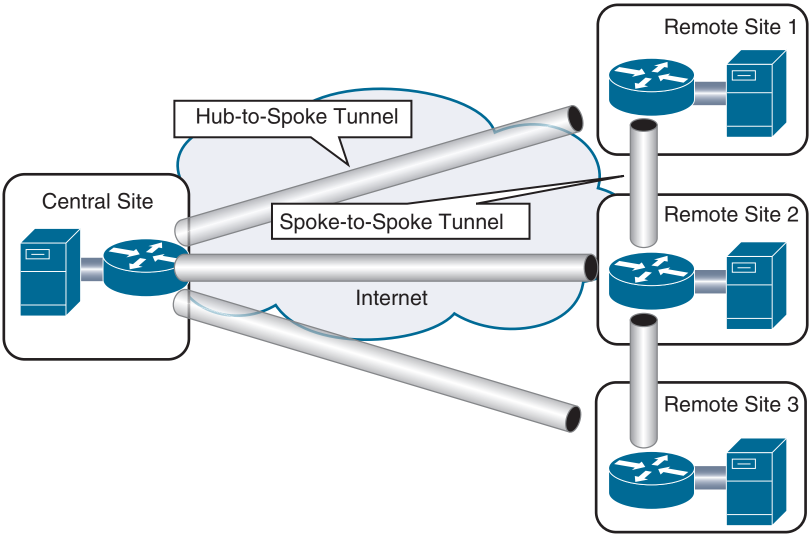
Cisco routers and Cisco ASA security appliances support site-to-site full-tunnel IPsec VPNs.

Dynamic Multipoint VPN

Dynamic Multipoint VPN (DMVPN) is a Cisco IOS Software solution for building scalable IPsec VPNs. DMVPN uses a centralized architecture to provide easier implementation and management for deployments that require granular access controls for diverse user communities, including mobile workers, telecommuters, and extranet users.

Cisco DMVPN allows branch locations to communicate directly with each other over the public WAN or Internet, such as when using VoIP between two branch offices, but does not require a permanent VPN connection between sites. It enables zero-touch deployment of IPsec VPNs and improves network performance by reducing latency and jitter, while optimizing head office bandwidth utilization. Figure 20-5 illustrates a simple DMVPN scenario with dynamic site-to-site tunnels being established from spokes to the hub or from spoke to spoke, as needed.

Figure 20-5 Cisco DMVPN Topology



Cisco IOS FlexVPN

Large customers deploying IPsec VPNs over IP networks are faced with high complexity and high costs for deploying multiple types of VPNs to meet different types of connectivity requirements. Customers often must learn different types of VPNs to manage and operate different types of networks. After an organization chooses a technology for a deployment, it typically tries to avoid migrating or adding functionality to enhance the VPN. Cisco FlexVPN was created to simplify the deployment of VPNs, to address the complexity of multiple solutions, and, as a unified ecosystem, to cover all types of VPNs, including remote access, teleworker, site-to-site, mobility, and managed security services.

As customer networks span private, public, and cloud systems, unifying the VPN technology becomes essential, and it is important to address the need for simplification of design and configuration. Customers can dramatically increase the reach of their network without significantly expanding the complexity of the infrastructure by using Cisco IOS FlexVPN. FlexVPN is a robust, standards-based encryption technology that helps enable large organizations securely connect branch offices and remote users and provides significant cost savings compared to supporting multiple separate types of VPN solutions, such as GRE (Generic Routing Encapsulation), crypto maps, and VTI-based solutions. FlexVPN relies on open-standards-based IKEv2 as a security technology and provides many Cisco enhancements to provide high levels of security.

FlexVPN can be deployed either over a public (Internet) network or a private MPLS (Multiprotocol Label Switching) VPN network. It is designed for concentration of both site-to-site VPNs and remote-access VPNs. One single FlexVPN deployment can accept both types of connection requests at the same time. Three different types of redundancy model can be implemented with FlexVPN: dynamic routing protocols over FlexVPN tunnels, IKEv2-based dynamic route distribution and server clustering, and IPsec/IKEv2 active/standby stateful failover between two chassis. FlexVPN natively supports IP multicast and QoS.

IPsec VPN Overview

IPsec, which is defined in RFC 4301, is designed to provide interoperable, high-quality, and cryptographically based transmission security to IP traffic. IPsec offers access control, connectionless integrity, data origin authentication, protection against replays, and confidentiality. These services are provided at the IP layer and offer protection for IP and upper-layer protocols.

IPsec combines the protocols IKE/IKEv2, Authentication Header (AH), and Encapsulation Security Payload (ESP) into a cohesive security framework.

IPsec provides security services at the IP layer by enabling a system that chooses required security protocols, determines the algorithm (or algorithms) to use for the service (or services), and puts in place any cryptographic keys that are required to provide the requested services. IPsec can protect one or more paths between a pair of hosts, between a pair of security gateways (usually routers or firewalls), or between a security gateway and a host.

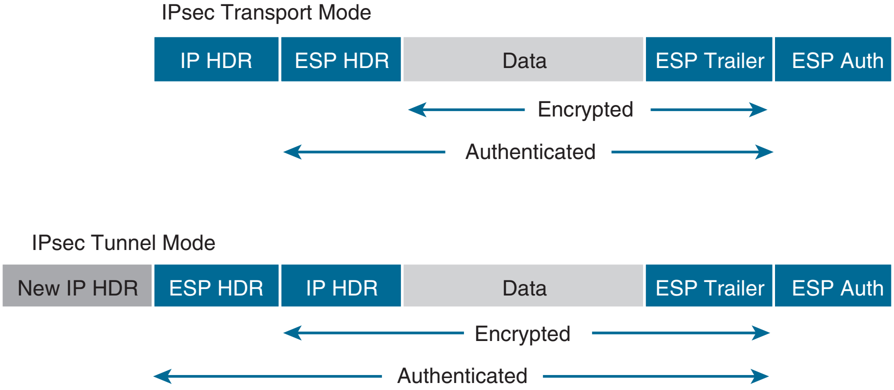
The IPsec protocol provides IP network layer encryption and defines a new set of headers to be added to IP datagrams. Two modes are available when implementing IPsec:

- **Transport mode:** This mode encrypts only the data portion (payload) of each packet and leaves the original IP packet header untouched. Transport mode is applicable to either gateway or host implementations, and it provides protection for upper-layer protocols and selected IP header fields.

- **Tunnel mode:** This mode is more secure than transport mode as it encrypts both the payload and the original IP header. IPsec in tunnel mode is normally used when the ultimate destination of a packet is different from the security termination point. This mode is also used in cases in which security is provided by a device that did not originate packets, such as with VPNs. Tunnel mode is often used in networks with unregistered IP addresses. The unregistered addresses can be tunneled from one gateway encryption device to another by hiding the unregistered addresses in the tunneled packet. Tunnel mode is the default for IPsec VPNs on Cisco devices.

Figure 20-6 illustrates the encapsulation process when transport mode and tunnel mode are used with ESP.

Figure 20-6 IPsec Transport and Tunnel Modes



IPsec also combines the following security protocols:

- IKE (Internet Key Exchange) provides key management to IPsec.
- AH (Authentication Header) defines a user traffic encapsulation that provides data integrity, data origin authentication, and protection against replay to user traffic. AH provides no encryption.
- ESP (Encapsulating Security Payload) defines a user traffic encapsulation that provides data integrity, data origin authentication, protection against replays, and confidentiality to user traffic. ESP offers data encryption and is preferred over AH.

You can use AH and ESP independently or together, although for most applications, just one of them is typically used. ESP is preferred, and AH is now considered obsolete and rarely used on its own.

IP Security Services

IPsec provides several essential security functions:

- **Confidentiality:** IPsec ensures confidentiality by using encryption. Data encryption prevents third parties from reading the data. Only the IPsec peer can decrypt and read the encrypted data.

- **Data integrity:** IPsec ensures that data arrives unchanged at the destination, meaning that the data has not been manipulated at any point along the communication path. IPsec ensures data integrity by using hash-based message authentication with MD5 or SHA.
- **Origin authentication:** Authentication ensures that the connection is made with the desired communication partner. Extended authentication can also be implemented to provide authentication of a user behind the peer system. IPsec uses IKE to authenticate users and devices that can carry out communication independently. IKE can use the following methods to authenticate the peer system:
    - Pre-shared keys (PSKs)
    - Digital certificates
        - RSA-encrypted nonces
- **Antireplay protection:** Antireplay protection verifies that each packet is unique and that packets are not duplicated. IPsec packets are protected by comparing the sequence number of the received packets with a sliding window on the destination host or security gateway. A packet that has a sequence number that comes before the sliding window is considered either late or a duplicate packet. Late and duplicate packets are dropped.
- **Key management:** Key management allows for an initial exchange of dynamically generated keys across a nontrusted network and a periodic rekeying process, limiting the maximum amount of time and the data that are protected with any one key.

The following are some of the encryption algorithms and key lengths that IPsec can use for confidentiality:

- **DES algorithm:** DES, which was developed by IBM, uses a 56-bit key to ensure high-performance encryption. DES is a symmetric key cryptosystem.
- **3DES algorithm:** The 3DES algorithm is a variant of 56-bit DES. 3DES operates in a way that is similar to how DES operates, in that data is broken into 64-bit blocks. 3DES processes each block three times, each time with an independent 56-bit key. 3DES provides a significant improvement in encryption strength over 56-bit DES. 3DES is a symmetric key cryptosystem. DES and 3DES should be avoided in favor of AES.
- **AES:** The National Institute of Standards and Technology (NIST) adopted AES to replace the aging DES-based encryption in cryptographic devices. AES provides stronger security than DES and is computationally more efficient than 3DES. AES offers three different key lengths: 128-, 192-, and 256-bit keys.
- **RSA:** RSA is an asymmetrical key cryptosystem. It commonly uses a key length of 1024 bits or larger. IPsec does not use RSA for data encryption. IKE uses RSA encryption only during the peer authentication phase.

Symmetric encryption algorithms such as AES require a common shared-secret key to perform encryption and decryption. You can use email, courier, or overnight express to send the shared-secret keys to the administrators of devices. This method is obviously impractical, and it does not guarantee that keys are not intercepted in transit. Public-key exchange methods, including the following methods, allow shared keys to be dynamically generated between the encrypting and decrypting devices:

- **Diffie-Hellman (DH):** The DH key agreement is a public-key exchange method that provides a way for two peers to establish a shared-secret key, which only they know, even though they are communicating over an insecure channel.
- **Elliptic Curve Diffie-Hellman (ECDH):** ECDH is a more secure variant of DH.

These algorithms are used within IKE to establish session keys. They support different prime sizes that are identified by different DH or ECDH groups. DH groups vary in the computational expense required for key agreement and strength against cryptographic attacks. Larger prime sizes provide stronger security but require more computational horsepower to execute. The following list shows the DH group numbers and their associated crypto strength:

- **DH1:** 768-bit
- **DH2:** 1024-bit
- **DH5:** 1536-bit
- **DH14:** 2048-bit
- **DH15:** 3072-bit
- **DH16:** 4096-bit
- **DH19:** 256-bit ECDH
- **DH20:** 384-bit ECDH
- **DH24:** 2048-bit ECDH

VPN data is transported over untrusted networks such as the public Internet. This data could potentially be intercepted and read or modified. To guard against this, IPsec uses HMACs.

IPsec uses Hashed-Based Message Authentication Code (HMAC) as the data integrity algorithm that verifies the integrity of a message. HMAC is defined in RFC 2104. HMAC utilizes a secret key known to the sender and the receiver. This is a lot like a keyed hash, but HMAC also adds padding logic and XOR logic, and it uses two hash calculations to produce the message authentication code.

When you are conducting business long distance, it is necessary to know who is at the other end of the phone, email, or fax. Similarly, with VPN networks, the device on the other end of the VPN tunnel must be authenticated before the communication path is considered secure. The following options are available for this authentication:

- **PSKs:** A secret key value is entered into each peer manually and is used to authenticate the peer. At each end, the PSK is combined with other information to form the authentication key.
- **RSA signatures:** The exchange of digital certificates authenticates the peers. The local device derives a hash and encrypts it with its private key. The encrypted hash is attached to the message and is forwarded to the remote end, and it acts like a signature. At the remote end, the encrypted hash is decrypted using the public key of the local end. If the decrypted hash matches the recomputed hash, the signature is genuine.

- **RSA encrypted nonces:** A nonce is a random number that is generated by the peer. RSA-encrypted nonces use RSA to encrypt the nonce value and other values. This method requires that each peer be aware of the public key of the other peer before negotiation starts. For this reason, public keys must be manually copied to each peer as part of the configuration process. This method is the least used of the authentication methods.

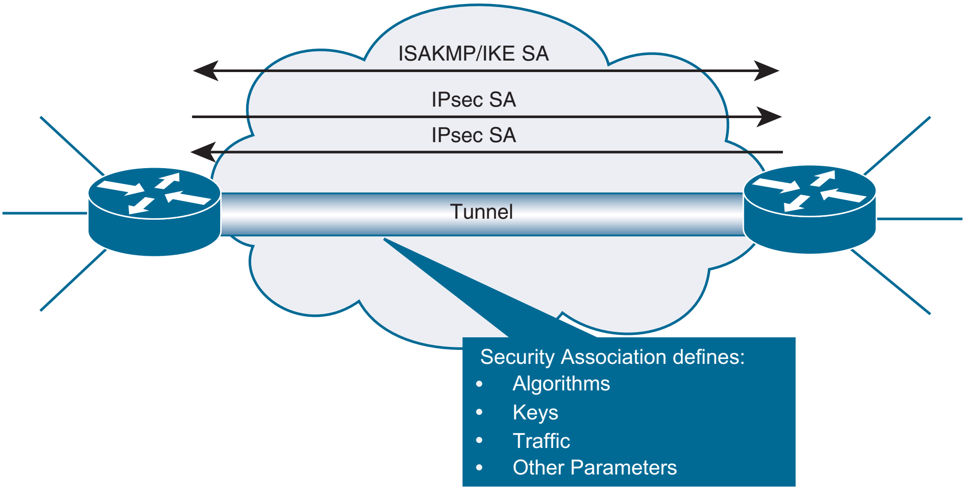
- **ECDSA signatures:** The Elliptic Curve Digital Signature Algorithm (ECDSA) is the elliptic curve analog of the DSA signature method. ECDSA signatures are smaller than RSA signatures of similar cryptographic strength. ECDSA public keys (and certificates) are smaller than similar-strength DSA keys, resulting in improved communications efficiency. Furthermore, on many platforms, ECDSA operations can be computed more quickly than similar-strength RSA operations. These advantages of signature size, bandwidth, and computational efficiency make ECDSA an attractive choice for many IKE and IKE Version 2 (IKEv2) implementations.

IPsec Security Associations

The concept of a security association (SA) is fundamental to IPsec. Both AH and ESP use security associations, and a major function of IKE is to establish and maintain security associations.

A security association is a simple description of current traffic protection parameters (algorithms, keys, traffic specification, and so on) that you apply to specific user traffic flows, as shown in Figure 20-7. AH or ESP provides security services to a security association. If AH or ESP protection is applied to a traffic stream, two (or more) security associations are created to provide protection to the traffic stream. To secure typical bidirectional communication between two hosts or between two security gateways, two security associations (one in each direction) are required.

Figure 20-7 IPsec Security Associations



IKE is a hybrid protocol that was originally defined by RFC 2409. It uses parts of several other protocols—Internet Security Association and Key Management Protocol (ISAKMP), Oakley, and SKEME—to automatically establish a shared security policy and authenticated keys for services that require keys, such as IPsec. IKE creates an authenticated, secure connection (defined by a separate IKE security association that is distinct from IPsec security associations) between two entities and then negotiates the security associations on behalf of the IPsec stack. This process requires that the two entities authenticate themselves to each other and establish shared session keys that IPsec encapsulations and algorithms use to transform plaintext user traffic into ciphertext. Note that Cisco IOS Software uses both *ISAKMP* and *IKE* to refer to the same thing. Although they are somewhat different, you can consider ISAKMP and IKE to be equivalent.

IPsec: IKE

IPsec uses the IKE protocol to negotiate and establish secured site-to-site or remote-access VPN tunnels. IKE is a framework provided by the Internet Security Association and Key Management Protocol (ISAKMP) and parts of two other key management protocols: Oakley and Secure Key Exchange Mechanism (SKEME). An IPsec peer accepting incoming IKE requests listens on UDP port 500.

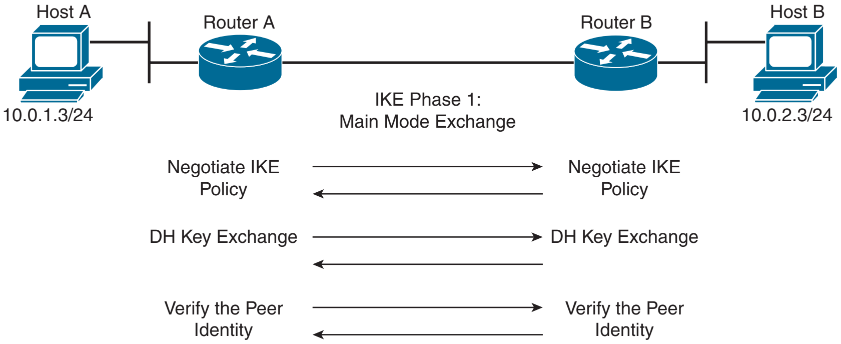
IKE uses ISAKMP for Phase 1 and Phase 2 of key negotiation. Phase 1 involves negotiating a security association (a key) between two IKE peers. The key negotiated in Phase 1 enables IKE peers to communicate securely in Phase 2. During Phase 2 negotiation, IKE establishes keys (security associations) for other applications, such as IPsec.

There are two versions of the IKE protocol: IKE Version 1 (IKEv1) and IKE Version 2 (IKEv2). IKEv2 was created to overcome some of the limitations of IKEv1. IKEv2 enhances the function of performing dynamic key exchange and peer authentication. It also simplifies the key exchange flows and introduces measures to fix vulnerabilities present in IKEv1. IKEv2 provides a simpler and more efficient exchange.

IKEv1 Phase 1

IKEv1 Phase 1 occurs in one of two modes: main mode or aggressive mode. Main mode has three two-way exchanges between the initiator and receiver. These exchanges define what encryption and authentication protocols are acceptable, how long keys should remain active, and whether perfect forward secrecy (PFS) should be enforced. IKE Phase 1 is illustrated in Figure 20-8.

Figure 20-8 IKEv1 Phase 1 Main Mode



The first step in IKEv1 main mode is to negotiate the security policy to use for the ISAKMP SA. There are five parameters, and they require agreement from both sides:

- Encryption algorithm
- Hash algorithm

- Diffie-Hellman group number
- Peer authentication method
- SA lifetime

The second exchange in IKEv1 main mode negotiations facilitates Diffie-Hellman key agreement. The Diffie-Hellman method allows two parties to share information over an untrusted network and mutually compute an identical shared secret that cannot be computed by eavesdroppers who intercept the shared information.

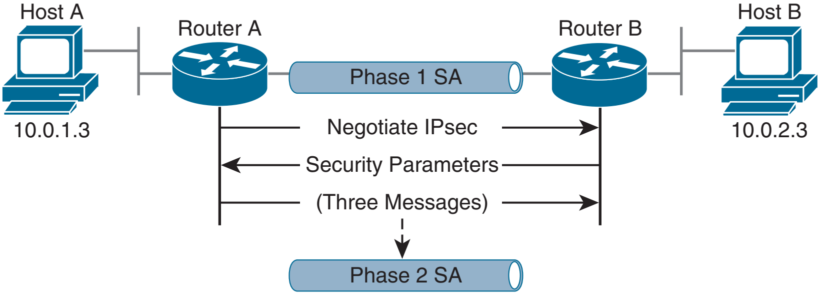
After the DH key exchange is complete, shared cryptographic keys are provisioned, but the peer is not yet authenticated. The device on the other end of the VPN tunnel must be authenticated before the communications path is considered secure. The last exchange of IKE Phase 1 authenticates the remote peer.

Aggressive mode, on the other hand, compresses the IKE SA negotiation phases that are described thus far into two exchanges and a total of three messages. In aggressive mode, the initiator passes all data required for the SA. The responder sends the proposal, key material, and ID and authenticates the session in the next packet. The initiator replies by authenticating the session. Negotiation is quicker, and the initiator and responder IDs pass in plaintext.

IKEv1 Phase 2

The purpose of IKE Phase 2 is to negotiate the IPsec security parameters that define the IPsec SA that protects the network data traversing the VPN. IKE Phase 2 offers only one mode, called quick mode, to negotiate the IPsec SAs. In Phase 2, IKE negotiates the IPsec transform set and the shared keying material that is used by the transforms. In this phase, the SAs that IPsec uses are unidirectional; therefore, a separate key exchange is required for each data flow. Optionally, Phase 2 can include its own Diffie-Hellman key exchange, using PFS. It is important to note that the ISAKMP SA in Phase 1 provides a bidirectional tunnel that is used to negotiate the IPsec SAs. Figure 20-9 illustrates the IKE Phase 2 exchange.

Figure 20-9 IKEv1 Phase 2



Quick mode typically uses three messages. For IKEv1 to create an IPsec security association using aggressive mode, a total of six messages are exchanged (three for aggressive mode and three for quick mode). If main mode is used, nine messages are exchanged (six for main mode and three for quick mode).

IKEv2

IKEv2 provides simplicity and increases speed by requiring fewer transactions to establish security associations. A simplified initial exchange of messages reduces latency and increases connection establishment speed. It incorporates many extensions that supplement the original IKE protocol, including NAT traversal, dead peer detection, and initial contact support. IKEv2 provides stronger security through DoS protection and other functions and provides reliability by using sequence numbers, acknowledgments, and error correction. It also provides flexibility, through support for EAP as a method for authenticating VPN endpoints. Finally, it provides mobility, by using the IKEv2 Mobility and Multihoming (MOBIKE) protocol extension. This enhancement allows mobile users to roam and change IP addresses without disconnecting their IPsec session.

IKEv2 reduces the number of exchanges from potentially six or nine messages down to four. IKEv2 has no option for either main mode or aggressive mode; there is only IKE_SA_INIT (security association initialization). Essentially, the initial IKEv2 exchange (IKE_SA_INIT) involves cryptographic algorithms and key material. So, the information exchanged in the first two pairs of messages in IKEv1 is exchanged in the first pair of messages in IKEv2. The next IKEv2 exchange (IKE_AUTH) is used to authenticate each peer and also to create a single pair of IPsec security associations. The information exchanged in the last two messages of main mode and in the first two messages of quick mode is exchanged in the IKE_AUTH exchange, in which both peers establish an authenticated, cryptographically protected IPsec security association. With IKEv2, all exchanges occur in pairs, and all messages sent require acknowledgment. If an acknowledgment is not received, the sender of the message is responsible for retransmitting it.

If additional IPsec security associations were required in IKEv1, a minimum of three messages would be used, and they would be created in quick mode. In contrast, IKEv2 employs just two messages with a CREATE_CHILD_SA exchange.

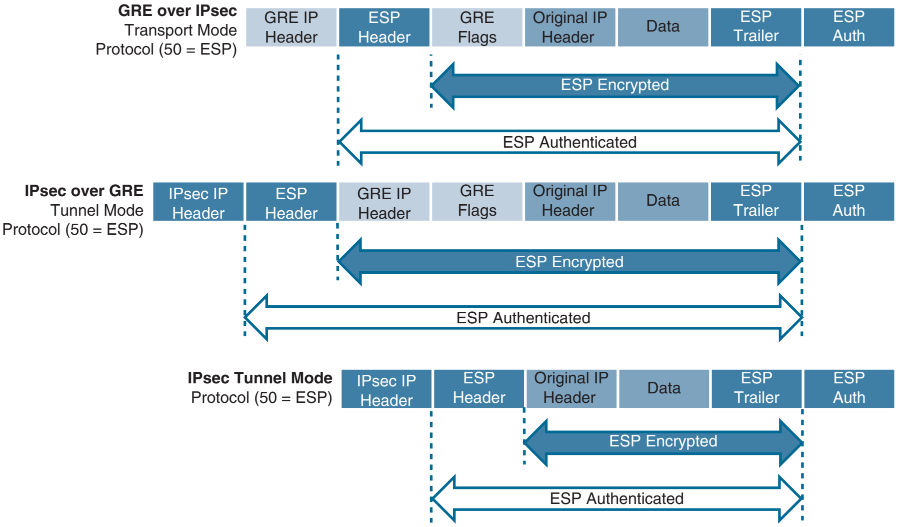
IKEv1 and IKEv2 are incompatible protocols, so you cannot configure an IKEv1 device to establish a VPN tunnel with an IKEv2 device.

IPsec Site-to-Site VPN Configuration

The GRE configuration shown in Example 20-1 allows for OSPF and user data traffic to flow between R1 and R4 encapsulated in a GRE packet. Since GRE traffic is neither encrypted nor authenticated, using GRE to carry confidential information across an insecure network like the Internet is not desirable. Instead, IPsec can be used to encrypt traffic traveling through a GRE tunnel. There are two combination options for IPsec and GRE to operate together, as shown in the first two packet encapsulation examples in Figure 20-10:

- **GRE over IPsec transport mode:** In this mode, the original packets are first encrypted and encapsulated into IPsec and then encapsulated within GRE. The GRE packet is then routed across the WAN using the GRE header.
- **IPsec over GRE tunnel mode:** In this mode, the original plaintext packet is encapsulated into GRE, and it contains the tunnel source and destination IP addresses. This is then protected by IPsec for confidentiality and/or integrity assurance, with an additional outer IP header to route the traffic to the destination.

Figure 20-10 GRE over IPsec vs. IPsec over GRE vs. IPsec Tunnel Mode


GRE over IPsec
Transport Mode
Protocol (50 = ESP)
GRE IP Header
ESP Header
GRE Flags
Original IP Header
Data
ESP Trailer
ESP Auth
ESP Encrypted
ESP Authenticated
IPsec over GRE
Tunnel Mode
Protocol (50 = ESP)
IPsec IP Header
ESP Header
GRE IP Header
GRE Flags
Original IP Header
Data
ESP Trailer
ESP Auth
ESP Encrypted
ESP Authenticated
IPsec Tunnel Mode
Protocol (50 = ESP)
IPsec IP Header
ESP Header
Original IP Header
Data
ESP Trailer
ESP Auth
ESP Encrypted
ESP Authenticated

Notice that when IPsec is combined with GRE, there is substantial header overhead, as a total of three IP headers are used with tunnel mode.

Another option is to use IPsec virtual tunnel interfaces (VTIs) instead. The use of IPsec VTIs simplifies the configuration process when you must provide protection for site-to-site VPN tunnels and offers a simpler alternative to the use of Generic Routing Encapsulation (GRE). A major benefit of IPsec VTIs is that the configuration does not require a static mapping of IPsec sessions to a physical interface. The IPsec tunnel endpoint is associated with a virtual interface. Because there is a routable interface at the tunnel endpoint, you can apply many common interface capabilities to the IPsec tunnel. Like GRE over IPsec, IPsec VTIs can natively support all types of IP routing protocols that provide scalability and redundancy. You can also use IPsec VTIs to securely transfer multicast traffic such as voice and video applications from one site to another. IPsec VTI tunnel mode encapsulation is shown at the bottom of Figure 20-10. Notice that there is no use of GRE in the encapsulation process, resulting in less header overhead.

The following sections look at both GRE over IPsec site-to-site VPNs using transport mode and VTI site-to-site VPNs.

GRE over IPsec Site-to-Site VPNs

There are two different ways to encrypt traffic over a GRE tunnel:

- Using IPsec crypto maps (legacy method)
- Using tunnel IPsec profiles (newer method)

The original implementation of IPsec VPNs used on Cisco IOS was known as *crypto maps*. The concept of configuring a crypto map was closely aligned to the IPsec protocol, and traffic that was required to be encrypted was defined in an access control list. This list was then referenced within an entry in the crypto map along with the IPsec cryptographic algorithms within the transform set. This configuration could become overly complex, and administrators introduced many errors when long access control lists were used.

Cisco introduced the concept of logical tunnel interfaces. These logical interfaces are basically doing the same as traditional crypto maps, but they are user configurable. The attributes used by this logical tunnel interface are referenced from the user-configured IPsec profile used to protect the tunnel. All traffic traversing this logical interface is protected by IPsec. This technique allows for traffic routing to be used to send traffic with the logical tunnel being the next hop, and it results in simplified configurations with greater flexibility for deployments.

Even though crypto maps are no longer recommended for tunnels, they are still widely deployed and should be understood.

GRE over IPsec Using Crypto Maps

Using the configuration in Example 20-1, which establishes a GRE tunnel between R1 and R4, follow these steps to enable IPsec on the GRE tunnel using crypto maps:

1. Define a crypto ACL to permit GRE traffic between the VPN endpoints R1 and R4, using the **access-list** *acl-number* **permit gre host** *tunnel-source-ip* **host** *tunnel-destination-ip* configuration command. This serves to define which traffic will be considered interesting for the tunnel. Notice that the ACLs on R1 and R4 are mirror images of each other.

2. Configure an ISAKMP policy for the IKE SA by using the **crypto isakmp policy** *priority* configuration command. Within the ISAKMP policy, configure the following security options:

   - Configure encryption (DES, 3DES, AES, AES-192, or AES-256) by using the **encryption** command

   - Configure a hash (SHA, SHA-256, SHA-384, or MD5) by using the **hash** command

   - Configure authentication (RSA signature, RSA encrypted nonce, or pre-shared key) by using the **authentication** command

   - Configure the Diffie-Hellman group (1, 2, 5, 14, 15, 16, 19, 20, or 24) by using the **group** command

3. Configure pre-shared keys (PSKs) by using the **crypto isakmp key** *key-string* **address** *peer-address* [*mask*] command. The same key needs to be configured on both peers, and the address 0.0.0.0 can be used to match all peers.

4. Create a transform set by using the **crypto ipsec transform-set** *transform-name* command. This command allows you to list a series of transforms to protect traffic flowing between peers. This step also allows you to configure either tunnel mode or transport mode. Recall that tunnel mode has extra IP header overhead compared to transport mode.

5. Build a crypto map by using the **crypto map** *map-name sequence-number* **ipsec-isakmp** command. Within the crypto map, configure the following security options:

   - Configure the peer IP address by using the **set peer** *ip-address* command

   - Configure the transform set to negotiate with the peer by using the **set transform-set** *transform-name* command

   - Configure crypto ACL to match by using the **match address** *acl-number* command

6. Apply the crypto map to the outside interface by using the **crypto map** *map-name* command.

Table 20-1 provides a side-by-side configuration that shows the commands used on R1 and R4 to establish a GRE over IPsec VPN using crypto maps. Notice that the IP addresses used in R1's configuration mirror those used on R4. (Refer to Figure 20-2 for the IP address information used in this example.)

Table 20-1 GRE over IPsec Configuration with Crypto Maps

R1

```
crypto isakmp policy 10
 hash sha256
 encryption aes 256
 authentication pre-share
 group 14
!
crypto isakmp key 31DAYS address
10.10.3.2
!
crypto ipsec transform-set MYSET
esp-aes 256 esp-sha-hmac
 mode transport
!
crypto map MYMAP 10 ipsec-isakmp
 set peer 10.10.3.2
 set transform-set MYSET
 match address 100
!
interface Tunnel0
 ip address 172.16.99.1 255.255.255.0
 tunnel source GigabitEthernet0/0
 tunnel destination 10.10.3.2
 ip mtu 1400
 bandwidth 1000
!
interface GigabitEthernet0/0
 ip address 10.10.1.1 255.255.255.0
 crypto map MYMAP
!
access-list 100 permit gre host
10.10.1.1 host 10.10.3.2
!
router ospf 1
 router-id 0.0.0.1
 network 172.16.99.0 0.0.0.255 area 0
 network 172.16.1.0 0.0.0.255 area 1
 network 172.16.11.0 0.0.0.255 area 1
```

R4

```
crypto isakmp policy 10
 hash sha256
 encryption aes 256
 authentication pre-share
 group 14
!
crypto isakmp key 31DAYS address
10.10.1.1
!
crypto ipsec transform-set MYSET
esp-aes 256 esp-sha-hmac
 mode transport
!
crypto map MYMAP 10 ipsec-isakmp
 set peer 10.10.1.1
 set transform-set MYSET
 match address 100
!
interface Tunnel0
 ip address 172.16.99.2 255.255.255.0
 tunnel source GigabitEthernet0/0
 tunnel destination 10.10.1.1
 ip mtu 1400
 bandwidth 1000
!
interface GigabitEthernet0/0
 ip address 10.10.3.2 255.255.255.0
 crypto map MYMAP
!
access-list 100 permit gre host
10.10.3.2 host 10.10.1.1
!
router ospf 1
 router-id 0.0.0.4
 network 172.16.99.0 0.0.0.255 area 0
 network 172.16.4.0 0.0.0.255 area 4
 network 172.16.14.0 0.0.0.255 area 4
```

GRE over IPsec Using Tunnel IPsec Profiles

Configuring a GRE over IPsec VPN using tunnel IPsec profiles instead of crypto maps requires the following steps:

1. Configure an ISAKMP policy for IKE SA. This step is identical to step 2 in the crypto maps example.
2. Configure PSKs. This step is identical to step 3 in the crypto maps example.

3. Create a transform set. This step is identical to step 4 in the crypto maps example.
4. Create an IPsec profile by using the **crypto ipsec profile** *profile-name* command. Associate the transform set configured in step 3 to the IPsec profile by using the **set transform-set** command.
5. Apply the IPsec profile to the tunnel interface by using the **tunnel protection ipsec profile** *profile-name* command.

Table 20-2 provides a side-by-side configuration that shows the commands used on R1 and R4 to establish a GRE over IPsec VPN using IPsec profiles. (Refer to Figure 20-2 for the IP address information used in this example.)

Table 20-2 GRE over IPsec Configuration with IPsec Profiles

R1

```
crypto isakmp policy 10
 hash sha256
 encryption aes 256
 authentication pre-share
 group 14
!
crypto isakmp key 31DAYS address
10.10.3.2
!
crypto ipsec transform-set MYSET
esp-aes 256 esp-sha-hmac
 mode transport
!
crypto ipsec profile MYPROFILE
 set transform-set MYSET
!
interface Tunnel0
 ip address 172.16.99.1
 255.255.255.0
 tunnel source GigabitEthernet0/0
 tunnel destination 10.10.3.2
 ip mtu 1400
 bandwidth 1000
 tunnel protection ipsec profile
MYPROFILE
!
router ospf 1
 router-id 0.0.0.1
 network 172.16.99.0 0.0.0.255 area 0
 network 172.16.1.0 0.0.0.255 area 1
 network 172.16.11.0 0.0.0.255 area 1
```

R4

```
crypto isakmp policy 10
 hash sha256
 encryption aes 256
 authentication pre-share
 group 14
!
crypto isakmp key 31DAYS address
10.10.1.1
!
crypto ipsec transform-set MYSET
esp-aes 256 esp-sha-hmac
 mode transport
!
crypto ipsec profile MYPROFILE
 set transform-set MYSET
!
interface Tunnel0
 ip address 172.16.99.2
 255.255.255.0
 tunnel source GigabitEthernet0/0
 tunnel destination 10.10.1.1
 ip mtu 1400
 bandwidth 1000
 tunnel protection ipsec profile
MYPROFILE
!
router ospf 1
 router-id 0.0.0.4
 network 172.16.99.0 0.0.0.255 area 0
 network 172.16.4.0 0.0.0.255 area 4
 network 172.16.14.0 0.0.0.255 area 4
```

Site-to-Site Virtual Tunnel Interface over IPsec

The steps to enable a VTI over IPsec are very similar to those for GRE over IPsec configuration using IPsec profiles. The only difference is the addition of the command **tunnel mode ipsec** {**ipv4** | **ipv6**} under the GRE tunnel interface to enable a VTI on it and to change the packet transport mode to tunnel mode. To revert to GRE over IPsec, you can use the command **tunnel mode gre** {**ip** | **ipv6**}.

Table 20-3 provides a side-by-side configuration that shows the commands used on R1 and R4 to establish a site-to-site VPN using VTI over IPsec. (Refer to Figure 20-2 for the IP address information used in this example.)

Table 20-3 VTI over IPsec Configuration

R1

```
crypto isakmp policy 10
 hash sha256
 encryption aes 256
 authentication pre-share
 group 14
!
crypto isakmp key 31DAYS address
10.10.3.2
!
crypto ipsec transform-set MYSET
esp-aes 256 esp-sha-hmac
 mode transport
!
crypto ipsec profile MYPROFILE
 set transform-set MYSET
!
interface Tunnel0
 ip address 172.16.99.1 255.255.255.0
 tunnel source GigabitEthernet0/0
 tunnel destination 10.10.3.2
 ip mtu 1400
 bandwidth 1000
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile
MYPROFILE
!
router ospf 1
 router-id 0.0.0.1
 network 172.16.99.0 0.0.0.255 area 0
 network 172.16.1.0 0.0.0.255 area 1
 network 172.16.11.0 0.0.0.255 area 1
```

R4

```
crypto isakmp policy 10
 hash sha256
 encryption aes 256
 authentication pre-share
 group 14
!
crypto isakmp key 31DAYS address
10.10.1.1
!
crypto ipsec transform-set MYSET
esp-aes 256 esp-sha-hmac
 mode transport
!
crypto ipsec profile MYPROFILE set
transform-set MYSET
!
interface Tunnel0
 ip address 172.16.99.2 255.255.255.0
 tunnel source GigabitEthernet0/0
 tunnel destination 10.10.1.1
 ip mtu 1400
 bandwidth 1000
 tunnel mode ipsec ipv4
 tunnel protection ipsec profile
MYPROFILE
!
router ospf 1
 router-id 0.0.0.4
 network 172.16.99.0 0.0.0.255 area 0
 network 172.16.4.0 0.0.0.255 area 4
 network 172.16.14.0 0.0.0.255 area 4
```

Example 20-3 shows the commands used to verify the status of the VTI IPsec tunnel between R1 and R4. The same commands can be used for the earlier example in which the IPsec tunnel was established by using crypto maps.

Example 20-3 Verifying VTI over IPsec

```
R1# show interface Tunnel 0
Tunnel0 is up, line protocol is up
  Hardware is Tunnel
  Internet address is 172.16.99.1/24
  MTU 17878 bytes, BW 1000 Kbit/sec, DLY 50000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel linestate evaluation up
  Tunnel source 10.10.1.1 (GigabitEthernet0/0), destination 10.10.3.2
   Tunnel Subblocks:
      src-track:
         Tunnel0 source tracking subblock associated with GigabitEthernet0/0
          Set of tunnels with source GigabitEthernet0/0, 1 member (includes
          iterators), on
          interface <OK>
  Tunnel protocol/transport IPSEC/IP
  Tunnel protection via IPSec (profile "MYPROFILE")
<. . . output omitted . . .>
```

```
R1# show crypto ipsec sa

interface: Tunnel0
    Crypto map tag: Tunnel0-head-0, local addr 172.16.99.1

   protected vrf: (none)
   local  ident (addr/mask/prot/port): (0.0.0.0/0.0.0.0/0/0)
   remote ident (addr/mask/prot/port): (0.0.0.0/0.0.0.0/0/0)
   current_peer 10.10.3.2 port 500
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 38, #pkts encrypt: 38, #pkts digest: 38
    #pkts decaps: 37, #pkts decrypt: 37, #pkts verify: 37
    #pkts compressed: 0, #pkts decompressed: 0
    #pkts not compressed: 0, #pkts compr. failed: 0
    #pkts not decompressed: 0, #pkts decompress failed: 0
    #send errors 0, #recv errors 0

     local crypto endpt.: 10.10.1.1, remote crypto endpt.: 10.10.3.2
     plaintext mtu 1438, path mtu 1500, ip mtu 1500, ip mtu idb GigabitEthernet0/0
     current outbound spi: 0xA3D5F191(2748707217)
     PFS (Y/N): N, DH group: none
```

```
      inbound esp sas:
       spi: 0x8A9B29A1(2325424545)
         transform: esp-256-aes esp-sha-hmac ,
         in use settings ={Transport, }
         conn id: 1, flow_id: SW:1, sibling_flags 80000040, crypto map:
         Tunnel0-head-0
         sa timing: remaining key lifetime (k/sec): (4608000/3101)
         IV size: 16 bytes
         replay detection support: Y
         Status: ACTIVE(ACTIVE)

      outbound esp sas:
       spi: 0x78A2BF51(2023931729)
         transform: esp-256-aes esp-sha-hmac ,
         in use settings ={Transport, }
         conn id: 2, flow_id: SW:2, sibling_flags 80000040, crypto map:
         Tunnel0-head-0
         sa timing: remaining key lifetime (k/sec): (4608000/3101)
         IV size: 16 bytes
         replay detection support: Y
         Status: ACTIVE(ACTIVE)
<. . . output omitted . . .>
```

```
R1# show crypto isakmp sa
IPv4 Crypto ISAKMP SA
dst              src              state           conn-id status
10.10.3.2        10.10.1.1        QM_IDLE            1008 ACTIVE
```

The **show interface Tunnel 0** command confirms the tunnel protocol in use (IPsec/IP) as well as the tunnel protection protocol (IPsec). The **show crypto ipsec sa** command displays traffic and VPN statistics for the IKE Phase 2 tunnel between R1 and R4. Notice the packets that were successfully encrypted and decrypted. Two SAs are established: one for inbound traffic and one for outbound traffic.

Finally, the **show crypto isakmp sa** command shows that the IKE Phase 1 tunnel is active between the two peers. QM_IDLE indicates that Phase 1 was successfully negotiated (either with main mode or aggressive mode) and that the ISAKMP SA is ready for use by quick mode in Phase 2.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	**Module, Chapter, or Link**
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	16
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	13
CCNP and CCIE Security Core SCOR 350-701 Official Cert Guide	8

Day 19

LISP and VXLAN

ENCOR 350-401 Exam Topics

Virtualization

- Describe network virtualization concepts
    - LISP
    - VXLAN

Key Topics

Today we review two important network overlay technologies: Locator/ID Separation Protocol (LISP) and Virtual Extensible Local Area Network (VXLAN). In the traditional Internet architecture, the IP address of an endpoint denotes both its location and its identity. Using the same value for both endpoint location and identity severely limits the security and management of traditional enterprise networks. LISP, defined in RFC 6830, is a protocol that enables separation of the endpoint's identity and its location.

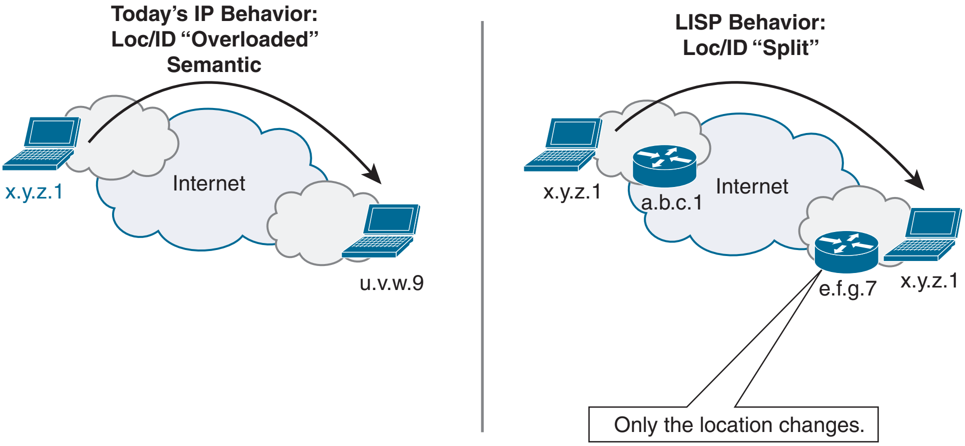
LISP has a limitation in that it supports only Layer 3 overlay. It cannot carry the MAC address because it discards the Layer 2 Ethernet header. In certain fabric technologies, such as SD-Access, the MAC address also needs to be carried; in such cases, VXLAN is deployed. RFC 7348 defines the use of VXLAN as a way to deploy a Layer 2 overlay network on top of a Layer 3 underlay network. VXLAN supports both Layer 2 and Layer 3 overlays. It preserves the original Ethernet header.

Locator/ID Separation Protocol

The creation of LISP was initially motivated by discussions during the IAB-sponsored Routing and Addressing Workshop held in Amsterdam in October 2006 (see RFC 4984). A key conclusion of the workshop was that the Internet routing and addressing system was not scaling well in the face of the explosive growth of new sites. One reason for this poor scaling was the increasing number of multihomed sites that could not be addressed as part of topology-based or provider-based aggregated prefixes.

In the current Internet routing and addressing architecture, a device's IP address is used as a single namespace that simultaneously expresses two functions of a device: its identity and how it is attached to the network. When that device moves, it must get a new IP address for both its identity and its location, as illustrated in the topology on the left in Figure 19-1.

Figure 19-1 IP Routing Model vs. LISP Routing Model



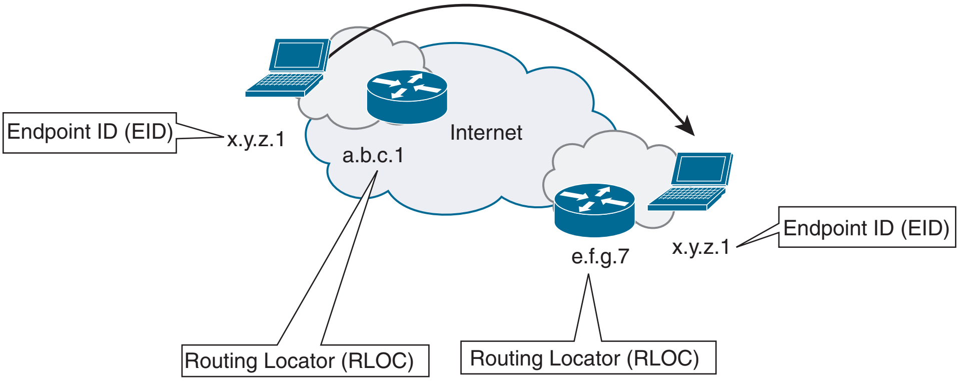
LISP is a routing and addressing architecture of the Internet Protocol. The LISP routing architecture was designed to solve issues related to scaling, multihoming, intersite traffic engineering, and mobility. An address on the Internet today combines location (how a device is attached to the network) and identity semantics in a single 32-bit (IPv4 address) or 128-bit (IPv6 address) number. LISP separates the location from the identity. In simple terms, with LISP, where you are (the network layer locator) in a network can change, but who you are (the network layer identifier) in the network remains the same. LISP separates the end-user device identifiers from the routing locators used by others to reach them.

When using LISP, a device's IP address represents only the device's identity. When the device moves, its IP address remains the same in both locations, and only the location ID changes, as show in the topology on the right in Figure 19-1.

The LISP routing architecture design creates a new paradigm, splitting the device identity from the location and defining two separate address spaces, as shown in Figure 19-2:

- **Endpoint identifier (EID) addresses:** These IP addresses and prefixes identify the endpoints or hosts. EID reachability across LISP sites is achieved by resolving EID-to-RLOC mappings.
- **Routing locator (RLOC) addresses:** These IP addresses and prefixes identify the different routers in the IP network. Reachability within the RLOC space is achieved by traditional routing methods.

Figure 19-2 LISP EID and RLOC Naming Convention

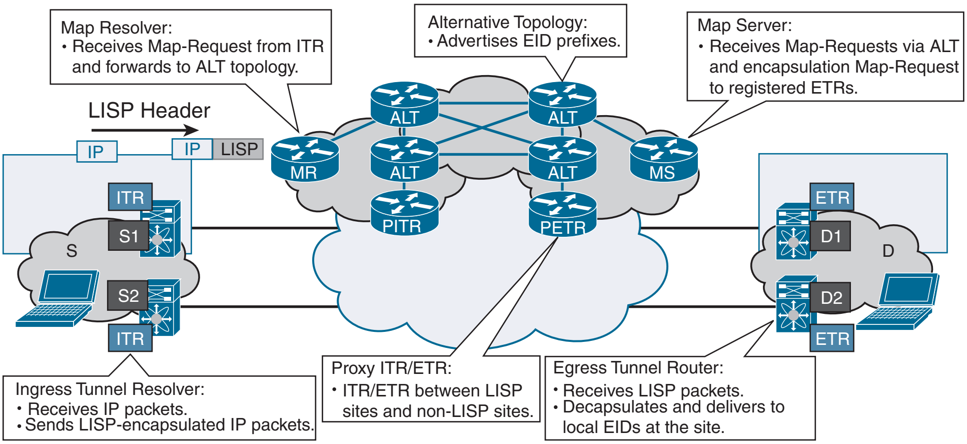

Endpoint ID (EID)
x.y.z.1
a.b.c.1
Internet
e.f.g.7
x.y.z.1
Endpoint ID (EID)
Routing Locator (RLOC)
Routing Locator (RLOC)

LISP uses a map-and-encapsulate routing model in which traffic that is destined for an EID is encapsulated and sent to an authoritative RLOC. This process is done rather than sending directly to the destination EID. It is based on the results of a lookup in a mapping database.

LISP Terms and Components

LISP uses a dynamic tunneling encapsulation approach rather than requiring preconfiguration of tunnel endpoints. It is designed to work in a multihoming environment, and it supports communication between LISP and non-LISP sites for interworking.

Figure 19-3 LISP Components


LISP Header
IP
IP
LISP
ITR
S1
S
S2
ITR
MR
ALT
ALT
PITR
ALT
ALT
PETR
MS
ETR
D1
D
D2
ETR

Map Resolver:
- Receives Map-Request from ITR and forwards to ALT topology.

Alternative Topology:
- Advertises EID prefixes.

Map Server:
- Receives Map-Requests via ALT and encapsulation Map-Request to registered ETRs.

Ingress Tunnel Resolver:
- Receives IP packets.
- Sends LISP-encapsulated IP packets.

Proxy ITR/ETR:
- ITR/ETR between LISP sites and non-LISP sites.

Egress Tunnel Router:
- Receives LISP packets.
- Decapsulates and delivers to local EIDs at the site.

LISP sites include the following devices, as illustrated in Figure 19-3:

- **Ingress tunnel router (ITR):** An ITR is a LISP site edge device that receives packets from site-facing interfaces (internal hosts) and encapsulates them to remote LISP sites or natively forwards them to non-LISP sites. An ITR is responsible for finding EID-to-RLOC mappings for all traffic destined for LISP-capable sites. When it receives a packet destined for an EID, it first looks for the EID in its mapping cache. If it finds a match, it encapsulates the packet inside a LISP header, with one of its RLOCs as the IP source address and one of the RLOCs from the mapping cache entry as the IP destination. It then routes the packet normally. If no entry is found in its mapping cache, the ITR sends a Map-Request message to one of its configured map resolvers. It then discards the original packet. When it receives a response to its Map-Request message, it creates a new mapping cache entry with the contents of the Map-Reply message. When another packet, such as a retransmission for the original, discarded packet arrives, the mapping cache entry is used for encapsulation and forwarding. Note that the Map-Reply message may indicate that the destination is not an EID; if that occurs, a negative mapping cache entry is created, which causes packets to either be discarded or forwarded natively when the cache entry is matched. The ITR function is usually implemented in the customer premises equipment (CPE) router. The same CPE router often provides both ITR and ETR functions; such a configuration is referred to as an xTR. In Figure 19-3, S1 and S2 are ITR devices.

- **Egress tunnel router (ETR):** An ETR is a LISP site edge device that receives packets from core-facing interfaces (the transport infrastructure), decapsulates LISP packets, and delivers them to local EIDs at the site. An ETR connects a site to the LISP-capable part of the Internet, publishes EID-to-RLOC mappings for the site, responds to Map-Request messages, and decapsulates and delivers LISP-encapsulated user data to end systems at the site. During operation, an ETR sends periodic Map-Register messages to all its configured map servers. The Map-Register messages contain all the EID-to-RLOC entries the ETR owns—that is, all the EID-numbered networks that are connected to the ETR's site. When an ETR receives a Map-Request message, it verifies that the request matches an EID for which it is responsible, constructs an appropriate Map-Reply message containing its configured mapping information, and sends this message to the ITR whose RLOCs are listed in the Map-Request message. When an ETR receives a LISP-encapsulated packet that is directed to one of its RLOCs, it decapsulates the packet, verifies that the inner header is destined for an EID-numbered end system at its site, and then forwards the packet to the end system using site-internal routing. Like the ITR function, the ETR function is usually implemented in a LISP site's CPE routers, typically as part of xTR function. In Figure 19-3, D1 and D2 are ETR devices

- **Map server (MS):** A LISP map server implements the mapping database distribution. It does this by accepting registration requests from its client ETRs, aggregating the EID prefixes that they successfully register, and advertising the aggregated prefixes to the ALT router with BGP. To do this, the MS is configured with a partial mesh of GRE tunnels and BGP sessions to other map server systems or ALT routers. Since a map server does not forward user data traffic, it does not have high-performance switching capability and is well suited for implementation on a general-purpose computing server rather than on special-purpose router hardware. Both map server and map resolver functions are typically implemented on a common system; such a system is referred to as a map resolver/map server (MR/MS).

- **Map resolver (MR):** Like a map server, a LISP map resolver connects to the ALT router by using a partial mesh of GRE tunnels and BGP sessions. It accepts encapsulated Map-Request messages sent by ITRs, decapsulates them, and then forwards them over the ALT router toward the ETRs responsible for the EIDs being requested.

- **Proxy ITR (PITR):** A PITR implements ITR mapping database lookups and LISP encapsulation functions on behalf of non-LISP-capable sites. PITRs are typically deployed near major Internet exchange points (IXPs) or in Internet service provider (ISP) networks to allow non-LISP customers of those facilities to connect to LISP sites. In addition to implementing ITR functions, a PITR also advertises some or all the nonroutable EID prefix space to the part of the non-LISP-capable Internet that it serves. This advertising is performed so that the non-LISP sites route traffic toward the PITR for encapsulation and forwarding to LISP sites. Note that these advertisements are intended to be highly aggregated, with many EID prefixes covered by each prefix advertised by a PITR.

- **Proxy ETR (PETR)**: A PETR implements ETR functions on behalf of non-LISP sites. A PETR is typically used when a LISP site needs to send traffic to non-LISP sites but cannot do so because its access network (the service provider to which it connects) does not accept nonroutable EIDs as packet sources. With dual-stacking, a PETR may also serve as a mechanism for LISP sites with EIDs within one address family and RLOCs within a different address family to communicate with each other. The PETR function is commonly offered by devices that also act as PITRs; such devices are referred to as PxTRs.

- **ALT router:** ALT routers may not be present in all mapping database deployments. When an ALT router is present, it connects through GRE tunnels and BGP sessions, map servers, map resolvers, and other ALT routers. Its only purpose is to accept EID prefixes advertised by devices that form a hierarchically distinct part of the EID numbering space and then advertise an aggregated EID prefix that covers that space to other parts of the ALT router. Just as in the global Internet routing system, such aggregation is performed to reduce the number of prefixes that need to be propagated throughout the entire network. A map server or a combined MR/MS may also perform such aggregation, thus implementing the functions of an ALT router.

The EID namespace is used within LISP sites for end-site addressing of hosts and routers. These EID addresses go in DNS records, as they do today. Generally, an EID namespace is not globally routed in the underlying transport infrastructure. RLOCs are used as infrastructure addresses for LISP routers and core routers (often belonging to service providers) and are globally routed in the underlying infrastructure, just as they are today. Hosts do not know about RLOCs, and RLOCs do not know about hosts.

LISP Data Plane

Figure 19-4 illustrates a LISP packet flow when the PC in the LISP site needs to reach a server at address 10.1.0.1 in the West-DC. The numerals in the figure correspond with the following steps:

1. The source endpoint (10.3.0.1), at a remote site, performs a DNS lookup to find the destination (10.1.0.1).

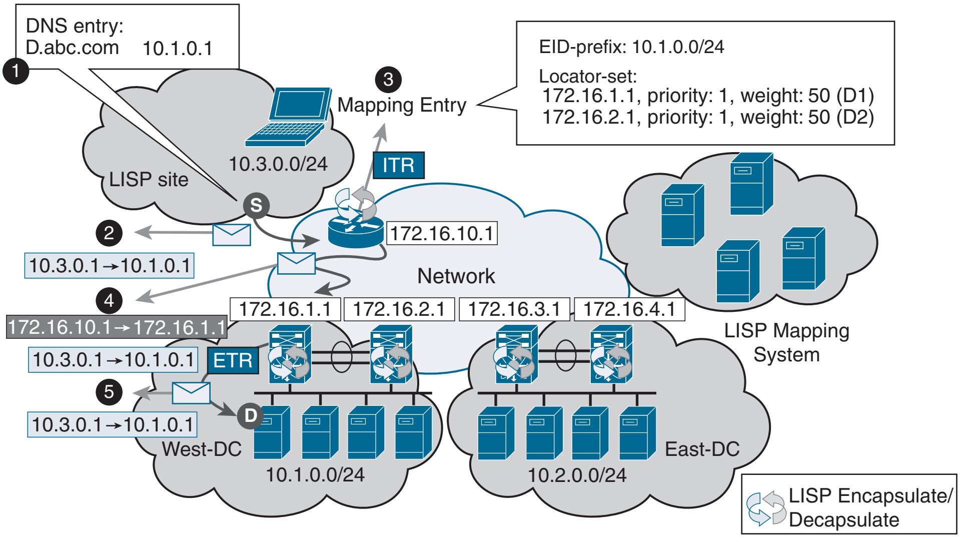
2. Traffic is remote, so it has to go through the branch router, which is a LISP-enabled device, in this scenario, playing the role of ITR.

3. The branch router does not know how to get to the specific address of the destination. It is LISP enabled, so it performs a LISP lookup to find a locator address. Notice that the destination EID subnet (10.1.0.1/24) is associated to the RLOCs (172.16.1.1 and 172.16.2.1)

identifying both ETR devices at the data center LISP-enabled site. Also, each entry has associated priority and weight values that the destination site uses to influence the way inbound traffic is received from the transport infrastructure. The priority is used to determine if both ETR devices can be used to receive LISP-encapsulated traffic that is destined to a local EID subnet (in a load-balancing scenario). The weight makes it possible to tune the amount of traffic that each ETR receives in a load-balancing scenario; therefore, the weight configuration makes sense only when specifying equal priorities for the local ETRs.

4. The ITR (branch router) performs an IP-in-IP encapsulation and transmits the data out the appropriate interface, based on standard IP routing decisions. The destination is one of the RLOCs of the data center ETRs. Assuming that the priority and weight values are configured the same on the ETR devices (as shown in Figure 19-4), the selection of the specific ETR RLOC is done on a per-flow basis, based on hashing that is performed on the Layer 3 and Layer 4 information of the IP packet of the original client.
5. The receiving LISP-enabled router receives the packet, decapsulates the packet, and forwards the packet to the final destination.

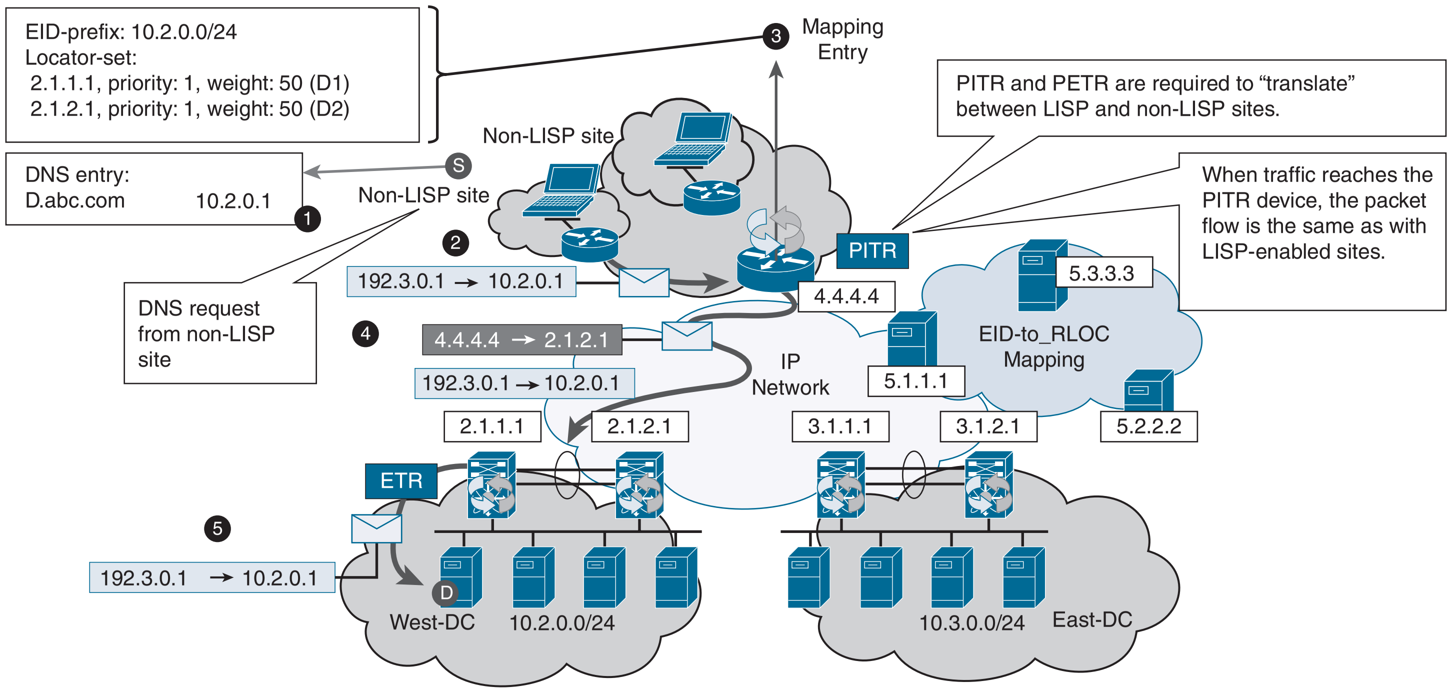
Figure 19-4 LISP Data Plane: LISP Site to LISP Site


DNS entry:
D.abc.com 10.1.0.1
1
3
Mapping Entry
EID-prefix: 10.1.0.0/24
Locator-set:
172.16.1.1, priority: 1, weight: 50 (D1)
172.16.2.1, priority: 1, weight: 50 (D2)
LISP site
10.3.0.0/24
ITR
S
2
172.16.10.1
10.3.0.1→10.1.0.1
Network
4
172.16.1.1
172.16.2.1
172.16.3.1
172.16.4.1
172.16.10.1→172.16.1.1
LISP Mapping System
10.3.0.1→10.1.0.1
ETR
5
D
10.3.0.1→10.1.0.1
West-DC
10.1.0.0/24
East-DC
10.2.0.0/24
LISP Encapsulate/ Decapsulate

A similar process occurs when a non-LISP site requires access to a LISP site. In Figure 19-5, the device at address 192.3.0.1 in the non-LISP site needs to reach a server at address 10.2.0.1 in the West-DC.

To fully implement LISP with Internet scale and interoperability between LISP and non-LISP sites, additional LISP infrastructure components are required to support the LISP-to-non-LISP interworking. These LISP infrastructure devices include the PITR and the PETR.

Figure 19-5 LISP Data Plane: Non-LISP Site to LISP Site


EID-prefix: 10.2.0.0/24
Locator-set:
2.1.1.1, priority: 1, weight: 50 (D1)
2.1.2.1, priority: 1, weight: 50 (D2)
DNS entry:
D.abc.com 10.2.0.1
1
S
3
Mapping Entry
Non-LISP site
Non-LISP site
DNS request from non-LISP site
2
192.3.0.1 → 10.2.0.1
4
4.4.4.4 → 2.1.2.1
192.3.0.1 → 10.2.0.1
PITR
4.4.4.4
IP Network
5.3.3.3
5.1.1.1
EID-to_RLOC Mapping
5.2.2.2
2.1.1.1
2.1.2.1
3.1.1.1
3.1.2.1
ETR
5
192.3.0.1 → 10.2.0.1
D
West-DC
10.2.0.0/24
10.3.0.0/24
East-DC

PITR and PETR are required to "translate" between LISP and non-LISP sites.

When traffic reaches the PITR device, the packet flow is the same as with LISP-enabled sites.

A proxy provides connectivity between non-LISP sites and LISP sites. The proxy functionality is a special case of ITR functionality in which the router attracts native packets from non-LISP sites (for example, the Internet) that are destined for LISP sites, and it encapsulates and forwards them to the destination LISP site.

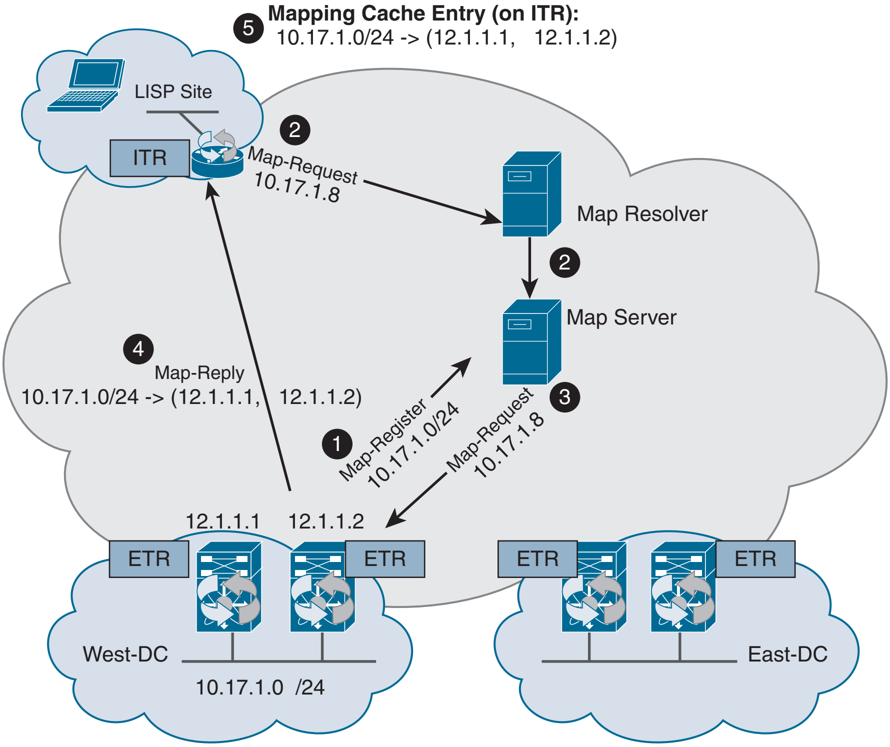
When the traffic reaches the PITR device, the mechanism that is used to send traffic to the EID in the data center is identical to what was previously discussed with a LISP-enabled remote site.

LISP is frequently used to steer traffic to and from data centers. It is common practice to deploy data centers in pairs to provide resiliency. When data centers are deployed in pairs, both facilities are expected to actively handle client traffic, and application workloads are expected to move freely between the data centers.

LISP Control Plane

The numerals in Figure 19-6 correspond with the following steps, which are required for an ITR to retrieve valid mapping information from the mapping database:

Figure 19-6 LISP Control Plane


⑤ Mapping Cache Entry (on ITR):
10.17.1.0/24 -> (12.1.1.1, 12.1.1.2)
LISP Site
ITR
② Map-Request 10.17.1.8
Map Resolver
②
Map Server
③ Map-Request 10.17.1.8
① Map-Register 10.17.1.0/24
④ Map-Reply
10.17.1.0/24 -> (12.1.1.1, 12.1.1.2)
12.1.1.1
12.1.1.2
ETR
ETR
ETR
ETR
West-DC
10.17.1.0 /24
East-DC

1. The ETRs register with the MS the EID subnets that are locally defined and that are authoritative. In this example, the EID subnet is 10.17.1.0/24. Every 60 seconds, each ETR sends a Map-Register message.

2. Assuming that a local map-cache entry is not available, when a client wants to establish communication to a data center EID, the remote ITR sends a Map-Request message to the map resolver, which then forwards the message to the map server.

3. The map server forwards the original Map-Request message to the ETR that last registered the EID subnet. In this example, it is ETR with locator 12.1.1.2.

4. The ETR sends to the ITR a Map-Reply message containing the requested mapping information.

5. The ITR installs the mapping information in its local map cache, and it starts encapsulating traffic toward the data center EID destination.

LISP Host Mobility

The decoupling of identity from the topology is the core principle on which the LISP host mobility solution is based. It allows the EID space to be mobile without impacting the routing that interconnects the locator IP space. When a move is detected, the mappings between EIDs and RLOCs are updated by the new xTR. By updating the RLOC-to-EID mappings, traffic is redirected to the new locations without requiring the injection of host routes or causing any churn in the underlying routing. In a virtualized data center deployment, EIDs can be directly assigned to virtual machines that are free to migrate between data center sites with their IP addressing information preserved.

LISP host mobility detects moves by configuring xTRs to compare the source in the IP header of traffic that is received from a host against a range of prefixes that are allowed to roam. These prefixes are defined as dynamic EIDs in the LISP host mobility solution. When deployed at the first-hop router (xTR), LISP host mobility devices also provide adaptable and comprehensive first-hop router functionality to service the IP gateway needs of the roaming devices that relocate.

LISP Host Mobility Deployment Models

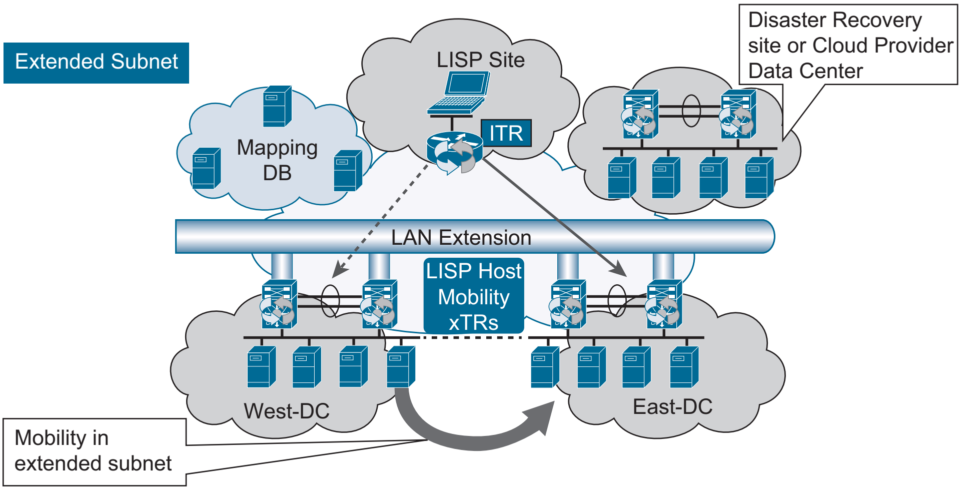
LISP host mobility offers two different deployment models, which are usually associated with the different workload mobility scenarios: LISP host mobility with an extended subnet and LISP host mobility across subnets.

LISP Host Mobility with an Extended Subnet

LISP host mobility with an extended subnet is usually deployed when geo-clustering or live workload mobility is required between data center sites so that the LAN extension technology provides the IP mobility functionality; LISP takes care of inbound traffic path optimization.

In Figure 19-7, a server is moved from West-DC to East-DC. The subnets are extended across the West-DC and East-DC using Virtual Private LAN Services (VPLS), Cisco Overlay Transport Virtualization (OTV), or something similar. In traditional routing, this would usually pose the challenge of steering the traffic originated from remote clients to the correct data center site where the workload is now located, given the fact that a specific IP subnet/VLAN is no longer associated with a single DC location. LISP host mobility is used to provide seamless ingress path optimization by detecting the mobile EIDs dynamically and updating the LISP mapping system with the current EID-RLOC mapping.

Figure 19-7 LISP Host Mobility in Extended Subnet


Extended Subnet
Mapping DB
LISP Site
ITR
Disaster Recovery site or Cloud Provider Data Center
LAN Extension
LISP Host Mobility xTRs
West-DC
East-DC
Mobility in extended subnet

LISP Host Mobility Across Subnets

The LISP host mobility across subnets model allows a workload to be migrated to a remote IP subnet while retaining its original IP address. You can generally use this model in cold migration scenarios (such as fast bring-up of disaster recovery facilities in a timely manner, cloud bursting, or data center migration/consolidation). In these use cases, LISP provides both IP mobility and inbound traffic path optimization functionalities.

In Figure 19-8, the LAN extension between West-DC and East-DC is still in place, but it is not deployed to the remote data center. A server is moved from East-DC to the remote data center. When the LISP VM router receives a data packet that is not from one of its configured subnets, it detects EIDs (VMs) across subnets. The LISP VM router then registers the new EID-to-RLOC mapping to the configured map servers associated with the dynamic EID.

LISP Host Mobility Example

Figure 19-9 illustrates a LISP host mobility example. The host (10.1.1.10/32) is connected to an edge device CE11 (12.1.1.1) in Campus Bldg 1. In the local routing table of edge device CE11, there is a host-specific entry for 10.1.1.10/32. Edge device CE11 registers the host with the map server. In the mapping database, you see that 10.1.1.10/32 is mapped to 12.1.1.1, which is the edge device CE11 in Campus Bldg 1. Traffic flows from source (10.1.1.10) to destination (10.10.10.0/24), based on the mapping entry.

Figure 19-8 LISP Host Mobility Across Subnets

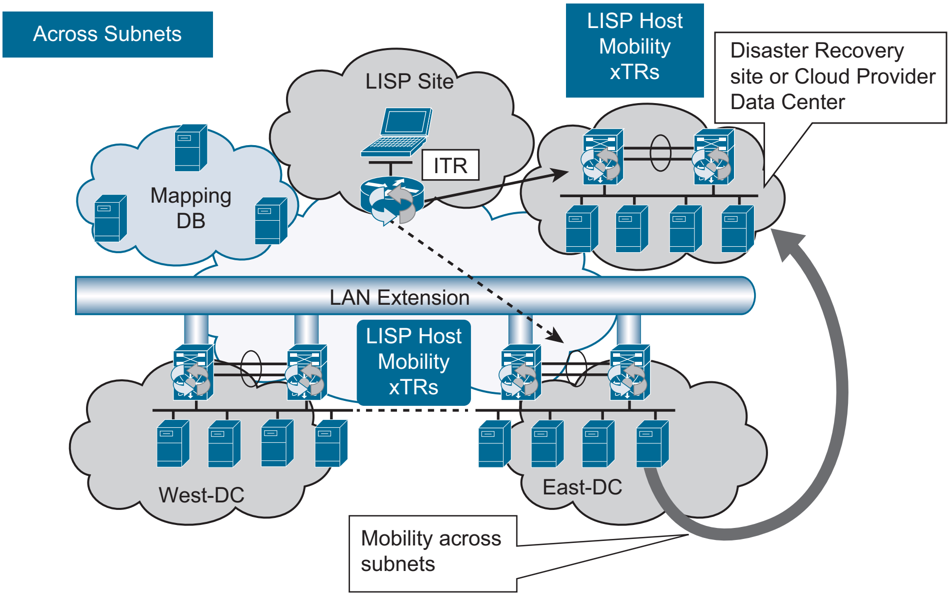

Across Subnets
LISP Site
ITR
LISP Host Mobility xTRs
Disaster Recovery site or Cloud Provider Data Center
Mapping DB
LAN Extension
LISP Host Mobility xTRs
West-DC
East-DC
Mobility across subnets

Figure 19-9 LISP Host Mobility Example: Before Host Migration

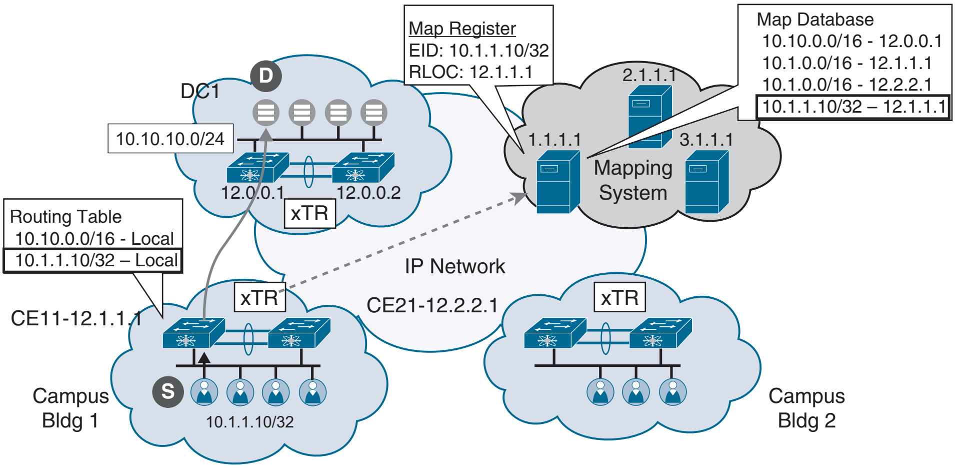
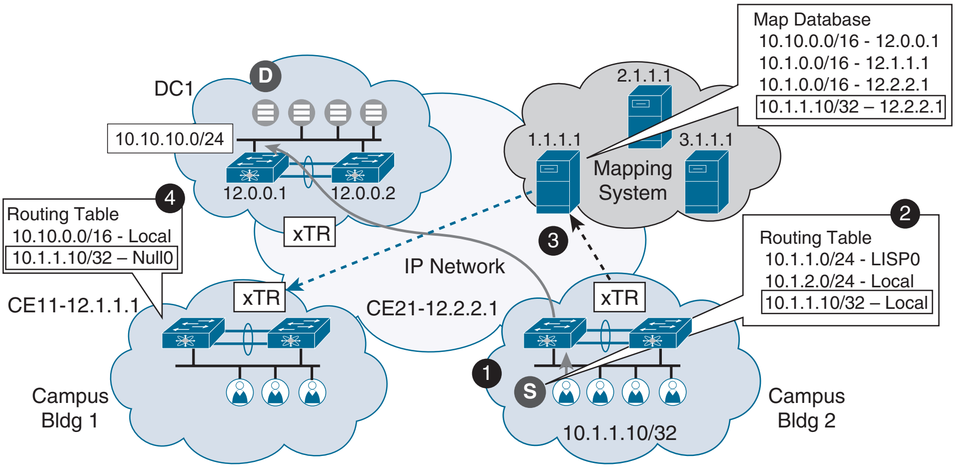

Map Register
EID: 10.1.1.10/32
RLOC: 12.1.1.1
Map Database
10.10.0.0/16 - 12.0.0.1
10.1.0.0/16 - 12.1.1.1
10.1.0.0/16 - 12.2.2.1
10.1.1.10/32 – 12.1.1.1
D
DC1
10.10.10.0/24
12.0.0.1
12.0.0.2
xTR
2.1.1.1
1.1.1.1
3.1.1.1
Mapping System
Routing Table
10.10.0.0/16 - Local
10.1.1.10/32 – Local
IP Network
CE11-12.1.1.1
CE21-12.2.2.1
xTR
xTR
S
Campus Bldg 1
10.1.1.10/32
Campus Bldg 2

Figure 19-10 shows what happens when the 10.1.1.10 host moves from Campus Bldg 1 to Campus Bldg 2. In this case, the 10.1.0.0/16 subnet is extended between Campus Bldg 1 and Campus Bldg 2. The numerals in the figure correspond to the following steps:

Figure 19-10 LISP Host Mobility Example: After Host Migration


DC1
D
10.10.10.0/24
12.0.0.1
12.0.0.2
xTR
Map Database
10.10.0.0/16 - 12.0.0.1
10.1.0.0/16 - 12.1.1.1
10.1.0.0/16 - 12.2.2.1
10.1.1.10/32 – 12.2.2.1
2.1.1.1
1.1.1.1
3.1.1.1
Mapping System
4
Routing Table
10.10.0.0/16 - Local
10.1.1.10/32 – Null0
3
2
Routing Table
10.1.1.0/24 - LISP0
10.1.2.0/24 - Local
10.1.1.10/32 – Local
IP Network
xTR
CE11-12.1.1.1
CE21-12.2.2.1
xTR
1
S
Campus Bldg 1
Campus Bldg 2
10.1.1.10/32

1. The host 10.1.1.10/32 connects to edge device CE21 with IP address 12.2.2.1 at Campus Bldg 2.
2. The edge device CE21 adds the host-specific entry to its local routing table.
3. The edge device CE21 sends a Map-Register message to update the mapping table on the map server. The map server updates the entry and maps the host 10.1.1.10 to edge device 12.2.2.1.
4. The map server sends a message to the edge device CE11 at Campus Bldg 1 (12.1.1.1) to say that its entry is no longer valid as the host has moved to a different location. The edge device CE11 (12.1.1.1) removes the entry from its local routing table by using a Null0 entry.

Traffic continues to flow from the source to the destination in the data center, as shown in the figure.

Virtual Extensible LAN (VXLAN)

Traditional Layer 2 network segmentation that VLANs provide has become a limiting factor in modern data center networks due to its inefficient use of available network links, rigid requirements on device placements, and limited scalability of a maximum of 4094 VLANs. VXLAN is designed to provide the same Layer 2 network services as VLAN but with greater extensibility and flexibility.

Compared to VLAN, VXLAN offers the following benefits:

- Multitenant segments can be flexibly placed throughout the data center. VXLAN extends Layer 2 segments over the underlay Layer 3 network infrastructure, crossing the traditional Layer 2 boundaries.

- VXLAN supports 16 million coexistent segments, which are uniquely identified by their VXLAN network identifiers (VNIs).
- Available network paths can be better utilized. Because VLAN uses STP, which blocks the redundant paths in a network, you may end up using only half of the network links. VXLAN packets are transferred through the underlying network based on the Layer 3 header and can take advantage of typical Layer 3 routing, ECMP, and link aggregation protocols to use all available paths.

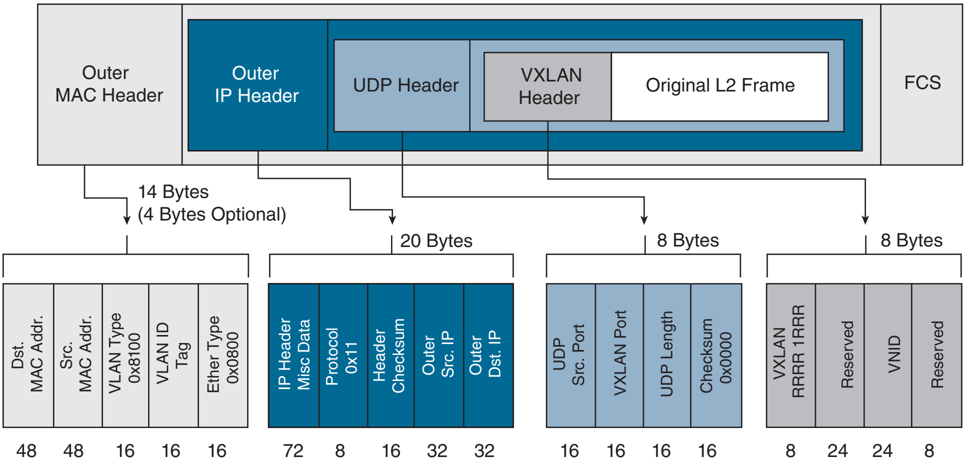
Because the overlay network is decoupled from the underlay network, it is considered flexible. Software-defined networking (SDN) controllers can reprogram the overlay network to suit the needs of a modern cloud platform. When used in an SDN environment like SD-Access, LISP operates at the control plane, while VXLAN operates at the data plane.

Both Cisco OTV and VXLAN technologies enable you to stretch a Layer 2 network. The primary difference between these two technologies is in usage. Cisco OTV is primarily used to provide Layer 2 connectivity over a Layer 3 network between two data centers. Cisco OTV uses mechanisms, such as ARP caching and IS-IS routing, to greatly reduce the amount of broadcast traffic; VXLAN is not as conservative because it is intended for use within a single data center.

VXLAN Encapsulation

VXLAN defines a MAC-in-UDP encapsulation scheme in which the original Layer 2 frame has a VXLAN header added and is then placed in a UDP IP packet. With this MAC-in-UDP encapsulation, VXLAN tunnels the Layer 2 network over the Layer 3 network. The VXLAN packet format is shown in Figure 19-11.

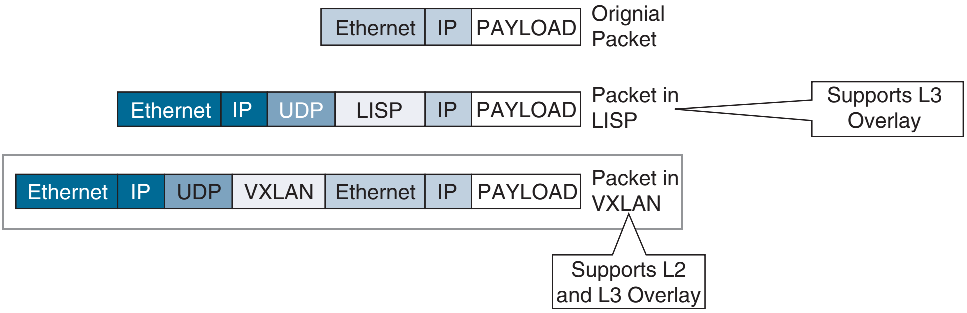
Figure 19-11 VXLAN Packet Format


Outer MAC Header
Outer IP Header
UDP Header
VXLAN Header
Original L2 Frame
FCS
14 Bytes (4 Bytes Optional)
20 Bytes
8 Bytes
8 Bytes
Dst. MAC Addr.
Src. MAC Addr.
VLAN Type 0x8100
VLAN ID Tag
Ether Type 0x0800
IP Header Misc Data
Protocol 0x11
Header Checksum
Outer Src. IP
Outer Dst. IP
UDP Src. Port
VXLAN Port
UDP Length
Checksum 0x0000
VXLAN RRRR 1RRR
Reserved
VNID
Reserved
48
48
16
16
16
72
8
16
32
32
16
16
16
16
8
24
24
8

As shown in Figure 19-11, VXLAN introduces an 8-byte VXLAN header that consists of a 24-bit VNI (VNID field) and a few reserved bits. The VXLAN header together with the original Ethernet frame goes in the UDP payload. The 24-bit VNI is used to identify Layer 2 segments and to maintain Layer 2 isolation between the segments. With all 24 bits in VNI, VXLAN can support 16 million LAN segments.

Figure 19-12 shows the relationship between LISP and VXLAN in the encapsulation process.

Figure 19-12 LISP and VXLAN Encapsulation


Ethernet
IP
PAYLOAD
Orignial Packet
Ethernet
IP
UDP
LISP
IP
PAYLOAD
Packet in LISP
Supports L3 Overlay
Ethernet
IP
UDP
VXLAN
Ethernet
IP
PAYLOAD
Packet in VXLAN
Supports L2 and L3 Overlay

When the original packet is encapsulated inside a VXLAN packet, the LISP header is preserved and used as the outer IP header. The LISP header carries a 24-bit Instance ID field that maps to the 24-bit VNID field in the VXLAN header.

VXLAN uses virtual tunnel endpoint (VTEP) devices to map devices in local segments to VXLAN segments. A VTEP performs encapsulation and decapsulation of the Layer 2 traffic. Each VTEP has at least two interfaces: a switch interface on the local LAN segment and an IP interface in the transport IP network, as illustrated in Figure 19-13.

Figure 19-13 VXLAN VTEP

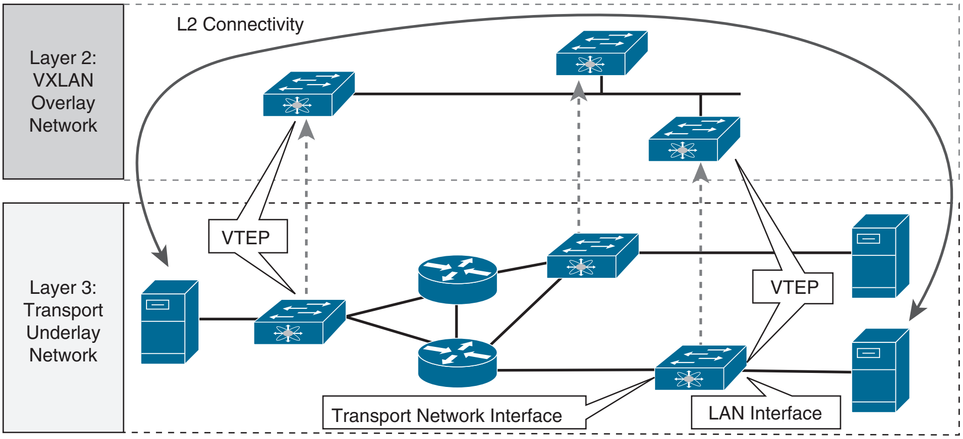
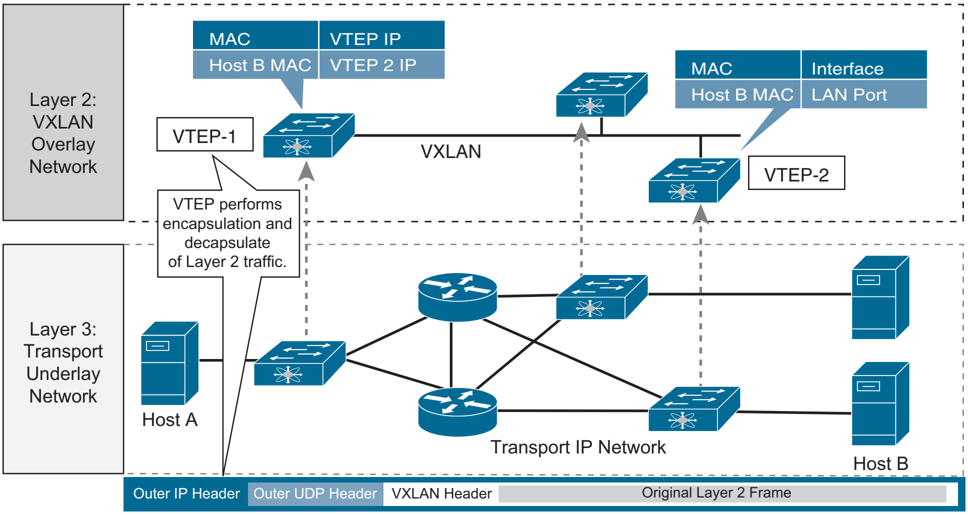

Layer 2: VXLAN Overlay Network
L2 Connectivity
Layer 3: Transport Underlay Network
VTEP
VTEP
Transport Network Interface
LAN Interface

Figure 19-14 demonstrates a VXLAN packet flow. When Host A sends traffic to Host B, it forms Ethernet frames with the MAC address for Host B as the destination MAC address and sends them to the local LAN. VTEP-1 receives the frame on its LAN interface. VTEP-1 has a mapping of MAC B to VTEP-2 in its VXLAN mapping table. It encapsulates the frames by adding a VXLAN header, a UDP header, and an outer IP address header to each frame using the destination IP address of

VTEP-2. VTEP-1 forwards the IP packets into the transport IP network based on the outer IP address header.

Figure 19-14 VXLAN Packet Flow


Layer 2: VXLAN Overlay Network
Layer 3: Transport Underlay Network
MAC | VTEP IP
Host B MAC | VTEP 2 IP
MAC | Interface
Host B MAC | LAN Port
VTEP-1
VTEP-2
VXLAN
VTEP performs encapsulation and decapsulate of Layer 2 traffic.
Host A
Host B
Transport IP Network
Outer IP Header
Outer UDP Header
VXLAN Header
Original Layer 2 Frame

Devices route packets toward VTEP-2 through the transport IP network. After VTEP-2 receives the packets, it strips off the outer Ethernet, IP, UDP, and VXLAN headers and forwards the packets through the LAN interface to Host B, based on the destination MAC address in the original Ethernet frame.

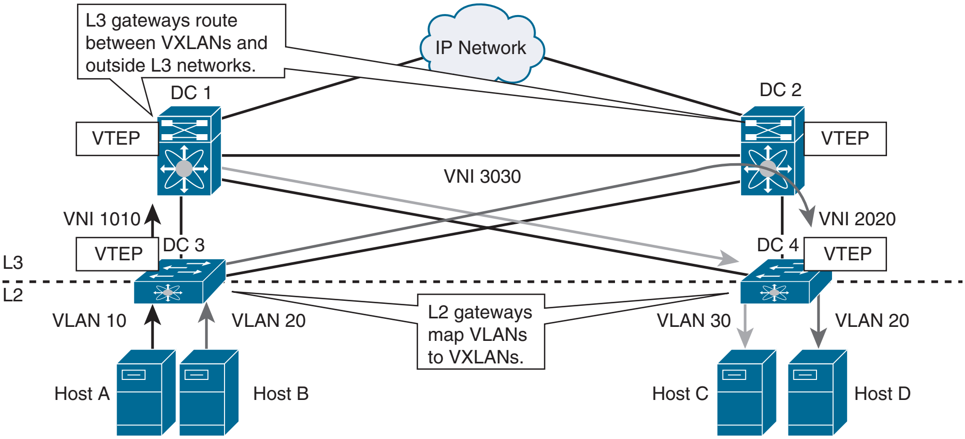
VXLAN Gateways

VXLAN is a relatively new technology, and data centers contain devices that are not capable of supporting VXLAN, such as legacy hypervisors, physical servers, and network services appliances. Those devices reside on classic VLAN segments. You enable VLAN-to-VXLAN connectivity by using a VXLAN Layer 2 gateway. A VXLAN Layer 2 gateway is a VTEP device that combines a VXLAN segment and a classic VLAN segment in one common Layer 2 domain.

Much as with traditional routing between different VLANs, a VXLAN Layer 3 gateway, also known as VXLAN router, routes between different VXLAN segments. The VXLAN router translates frames from one VNI to another. Depending on the source and destination, this process might require decapsulation and re-encapsulation of a frame. You could also implement routing between native Layer 3 interfaces and VXLAN segments.

Figure 19-15 illustrates a simple data center network with both VXLAN Layer 2 and Layer 3 gateways.

Figure 19-15 VXLAN Gateways


L3 gateways route between VXLANs and outside L3 networks.
IP Network
DC 1
DC 2
VTEP
VTEP
VNI 3030
VNI 1010
VNI 2020
DC 3
DC 4
VTEP
VTEP
L3
L2
VLAN 10
VLAN 20
L2 gateways map VLANs to VXLANs.
VLAN 30
VLAN 20
Host A
Host B
Host C
Host D

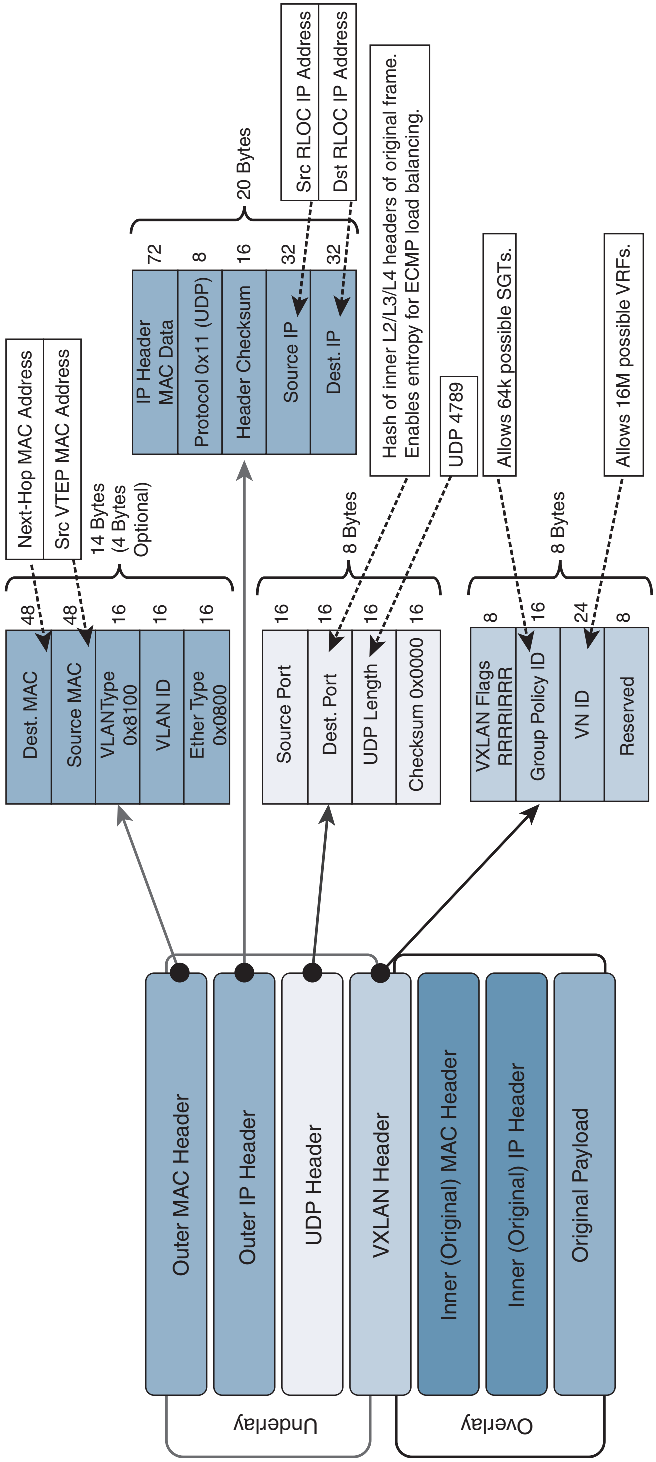
VXLAN-GPO Header

VXLAN Group Policy Option (VXLAN-GPO) is the latest version of VXLAN. It adds a special field in the header called Group Police ID to carry the Scalable Group Tags (SGTs). The outer part of the header consists of the IP and MAC addresses. It uses a UDP header with source and destination ports. The source port is a hash value that is created using the original source information and prevents polarization in the underlay. The destination port is always 4789. The frame can be identified as a VXLAN frame using the specific UDP designation port number.

Each overlay network is called a VXLAN segment; these segments are identified using 24-bit VXLAN virtual network IDs. The campus fabric uses the VXLAN data plane to provide transport of complete original Layer 2 frames and also uses LISP as the control plane to resolve endpoint-to-VTEP mappings. The campus fabric replaces 16 of the reserved bits in the VXLAN header to transport up to 64,000 SGTs. The virtual network ID maps to a VRF instance and enables the mechanism to isolate the data plane and the control plane across different virtual networks. The SGT carries user group membership information and is used to provide data plane segmentation inside the virtualized network.

Figure 19-16 shows the combination of underlay and overlay headers used in VXLAN-GPO. Notice that the outer MAC header carries VXLAN VTEP information, and the outer IP header carries LISP RLOC information.

Figure 19-16 VXLAN-GPO Header Fields


Underlay
Overlay
Outer MAC Header
Outer IP Header
UDP Header
VXLAN Header
Inner (Original) MAC Header
Inner (Original) IP Header
Original Payload
Dest. MAC
48
Source MAC
48
VLANType 0x8100
16
VLAN ID
16
Ether Type 0x0800
16
14 Bytes (4 Bytes Optional)
Next-Hop MAC Address
Src VTEP MAC Address
IP Header MAC Data
72
Protocol 0x11 (UDP)
8
Header Checksum
16
Source IP
32
Dest. IP
32
20 Bytes
Src RLOC IP Address
Dst RLOC IP Address
Source Port
16
Dest. Port
16
UDP Length
16
Checksum 0x0000
16
8 Bytes
Hash of inner L2/L3/L4 headers of original frame. Enables entropy for ECMP load balancing.
UDP 4789
VXLAN Flags RRRRIRRR
8
Group Policy ID
16
VN ID
24
Reserved
8
8 Bytes
Allows 64k possible SGTs.
Allows 16M possible VRFs.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	16
LISP Network Deployment and Troubleshooting: The Complete Guide to LISP Implementation on IOS-XE, IOS-XR, and NX-OS	1, 2
The LISP Network: Evolution to the Next Generation of Data Networks	2
Locator/ID Separation Protocol Architecture	https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/locator-id-separation-protocol-lisp/white_paper_c11-652502.html
CCNP and CCIE Data Center Core DCCOR 350-601 Official Cert Guide	Chapter 3
VXLAN Overview: Cisco Nexus 9000 Series Switches	https://www.cisco.com/c/en/us/products/collateral/switches/nexus-9000-series-switches/white-paper-c11-729383.html

Day 18

SD-Access

ENCOR 350-401 Exam Topics

Architecture

- Explain the working principles of the Cisco SD-Access solution
  - SD-Access control and data planes elements
  - Traditional campus interoperating with SD-Access

Key Topics

Today we review the first of two Cisco software-defined networking (SDN) technologies: Cisco Software-Defined Access (SD-Access). (The second Cisco SDN technology, Cisco SD-WAN, is covered on Day 17, "SD-WAN.") Cisco SD-Access is the evolution from traditional campus LAN designs to networks that directly implement the intent of an organization. SD-Access is enabled with an application package that runs as part of the Cisco Digital Network Architecture (DNA) Center software for designing, provisioning, applying policy, and facilitating the creation of an intelligent campus wired and wireless network with assurance.

Fabric technology, an integral part of Cisco SD-Access, provides wired and wireless campus networks with programmable overlays and easy-to-deploy network virtualization, permitting a physical network to host one or more logical networks, as required, to meet the design intent. In addition to network virtualization, fabric technology in the campus network enhances control of communications, providing software-defined segmentation and policy enforcement based on user identity and group membership. Software-defined segmentation is seamlessly integrated using Cisco TrustSec technology, providing microsegmentation for scalable groups in a virtual network using Scalable Group Tags (SGTs). Using Cisco DNA Center to automate the creation of virtual networks reduces operational expenses and also provides advantages such as reduced risk, integrated security, and improved network performance, thanks to the assurance and analytics capabilities.

Software-Defined Access

With the ever-growing needs of modern networks, the traditional methods of management and security have become challenging. New methods of device management and security configuration have been developed to ease the management overhead and reduce troubleshooting time and network outages. The Cisco SD-Access solution helps campus network administrators manage and secure the network by providing automation and assurance and reducing the burden and cost that traditional networks require.

Need for Cisco SD-Access

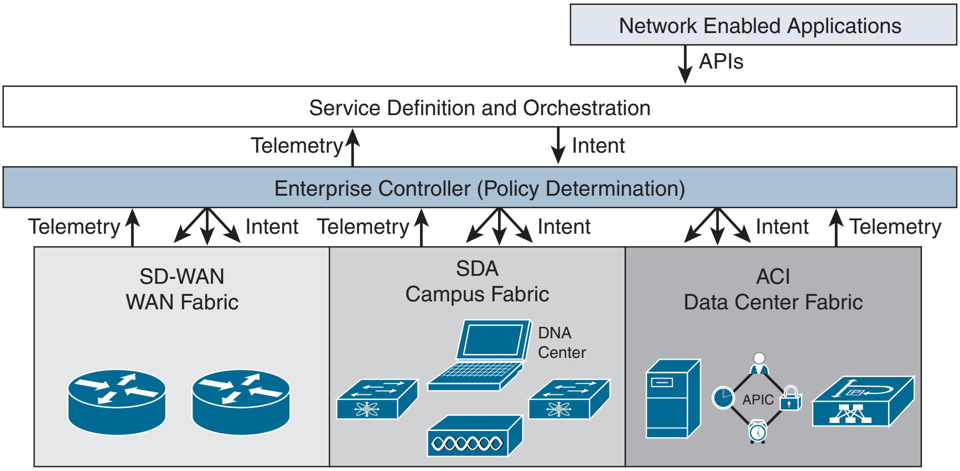
The Cisco Software-Defined Access (SD-Access) solution represents a fundamental change in the way to design, provision, and troubleshoot enterprise campus networks. Today, there are many challenges in managing the network to drive business outcomes. These limitations are due to manual configuration and fragmented tool offerings. There is high operational cost associated with implementing a fully segmented, policy-aware fabric architecture. In addition, manual configuration leads to higher network risk due to errors. Regulatory pressure is increasing due to escalating number of data breaches across the industry. More time is spent on troubleshooting the network than ever before due to the lack of network visibility and analytics.

Cisco SD-Access overcomes these challenges and provides the following benefits:

- A transformational management solution that reduces operational expenses (OpEx) and improves business agility
- Consistent management of wired and wireless networks from provisioning and policy perspectives
- Automated network segmentation and group-based policy
- Contextual insights for faster issue resolution and better capacity planning
- Open and programmable interfaces for integration with third-party solutions

Cisco SD-Access is part of the larger Cisco Digital Network Architecture (Cisco DNA). Cisco DNA also includes Cisco Software-Defined WAN (SD-WAN) and the data center Cisco Application Centric Infrastructure (ACI), as illustrated in Figure 18-1. We will discuss Cisco SD-WAN on Day 17. Cisco ACI is beyond the scope of the ENCOR exam.

Figure 18-1 Cisco DNA


Network Enabled Applications
APIs
Service Definition and Orchestration
Telemetry
Intent
Enterprise Controller (Policy Determination)
Telemetry
Intent
Telemetry
Intent
Intent
Telemetry
SD-WAN
WAN Fabric
SDA
Campus Fabric
DNA
Center
ACI
Data Center Fabric
APIC

Notice that each component of Cisco DNA relies on building and using a network fabric. Cisco SD-Access builds a standards-based network fabric that converts a high-level business policy into network configuration. The networking approach that is used to build the Cisco SD-Access fabric consists of an automatic physical underlay and a programmable overlay with constructs such as virtual networks and segments that can be further mapped to neighborhoods and groups of users. These constructs provide macro and micro segmentation capabilities to the network. SD-Access can be used to implement policy by mapping neighborhoods and groups of users to virtual networks and segments. This new approach enables enterprise networks to transition from traditional VLAN-centric design architecture to a new user group–centric design architecture.

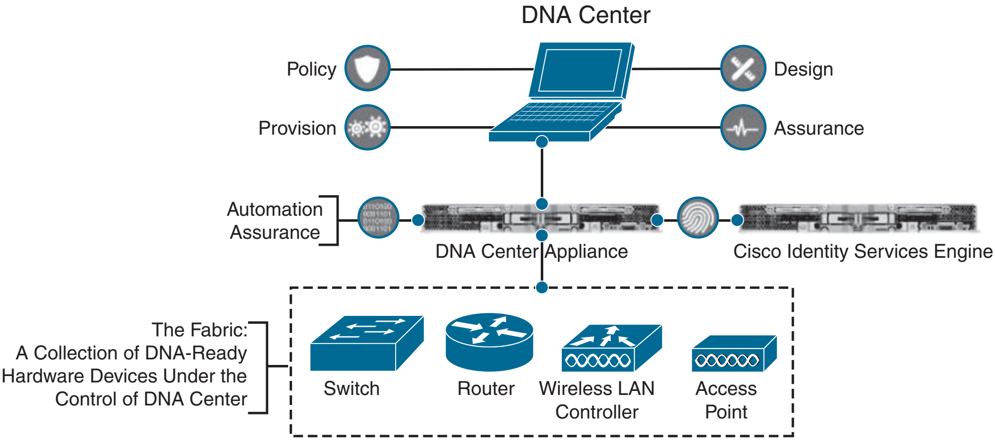
The Cisco SD-Access architecture offers simplicity with an open and standards-based API. Its simple user interface and native third-party app hosting enable easy orchestration with objects and data models. Automation and simplicity result in increased productivity. These benefits enable IT to be an industry leader in transforming a digital enterprise and providing consumers the ability to achieve operational effectiveness.

Enterprise networks have traditionally been configured using the CLI, and the same process had to be repeated each time a new site was brought up. This legacy network management is hardware-centric, requiring manual configurations, and uses script maintenance in a static environment, resulting in a slow workload change. This process is tedious and cannot scale in the new era of digitization, where network devices need to be provisioned and deployed quickly and efficiently.

Cisco SD-Access uses the new Cisco DNA Center, which was built on the Cisco Application Policy Infrastructure Controller Enterprise Module (APIC-EM). The Cisco DNA Center controller provides a single dashboard for managing your enterprise network. It uses intuitive workflows to simplify provisioning of user access policies that are combined with advanced assurance capabilities. It monitors the network proactively by gathering and processing information from devices, applications, and users. It identifies root causes and provides suggested remediation for faster troubleshooting. Machine learning continuously improves network intelligence to predict problems before they occur. This type of software-defined access control provides consistent policy and management across both wired and wireless segments, enables optimal traffic flows with seamless roaming, and allows an administrator to find any user or device on the network.

Figure 18-2 illustrates the relationship between Cisco DNA Center and the fabric technologies Cisco SD-Access and Cisco SD-WAN. Cisco Identity Services Engine (ISE) is an integral part of Cisco SD-Access for policy implementation, enabling dynamic mapping of users and devices to scalable groups and simplifying end-to-end security policy enforcement.

Figure 18-2 Cisco DNA Center


DNA Center
Policy
Provision
Design
Assurance
Automation
Assurance
DNA Center Appliance
Cisco Identity Services Engine
The Fabric:
A Collection of DNA-Ready
Hardware Devices Under the
Control of DNA Center
Switch
Router
Wireless LAN
Controller
Access
Point

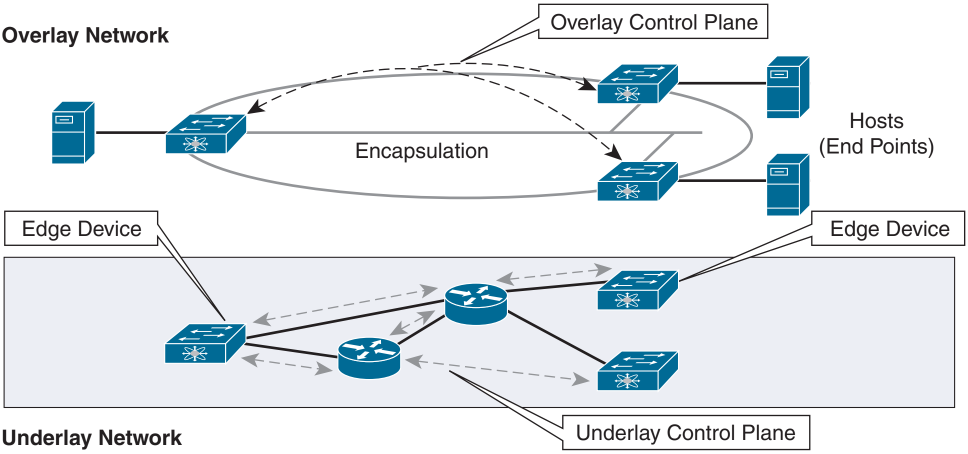
Cisco SD-Access Overview

The campus fabric architecture enables the use of virtual networks (overlay networks) that are running on a physical network (underlay network) to create alternative topologies to connect devices. Overlay networks are commonly used to provide Layer 2 and Layer 3 logical networks with virtual machine mobility in data center fabrics (for example, ACI, VXLAN, and FabricPath) and also in WANs to provide secure tunneling from remote sites (for example, MPLS, DMVPN, and GRE).

Cisco SD-Access Fabric

A fabric is an overlay. An overlay network is a logical topology that is used to virtually connect devices and is built on top of some arbitrary physical underlay topology. An overlay network often uses alternate forwarding attributes to provide additional services that are not provided by the underlay. Figure 18-3 illustrates the difference between the underlay network and an overlay network:

Figure 18-3 Overlay vs. Underlay Networks


Overlay Network
Overlay Control Plane
Encapsulation
Hosts
(End Points)
Edge Device
Edge Device
Underlay Control Plane
Underlay Network

- **Underlay network:** The underlay network is defined by the physical switches and routers that are parts of the campus fabric. All network elements of the underlay must establish IP connectivity via the use of a routing protocol. Theoretically, any topology and routing protocol can be used, but the implementation of a well-designed Layer 3 foundation to the campus edge is highly recommended to ensure performance, scalability, and high availability of the network. In the campus fabric architecture, end-user subnets are not part of the underlay network.
- **Overlay network:** An overlay network runs on top of the underlay to create a virtualized network. Virtual networks isolate both data plane traffic and control plane behavior among the virtualized networks from the underlay network. Virtualization is achieved inside the campus fabric by encapsulating user traffic over IP tunnels that are sourced and terminated at the boundaries of the campus fabric. The fabric boundaries include borders for ingress and egress to a fabric, fabric edge switches for wired clients, and fabric APs for wireless clients. Network virtualization extending outside the fabric is preserved using traditional virtualization technologies such as VRF-Lite and MPLS VPN. Overlay networks can run across all or a subset of the underlay network devices. Multiple overlay networks can run across the same underlay network to support multitenancy through virtualization.

The role of the underlay network is to establish physical connectivity from one edge device to another. It uses a routing protocol and a distinct control plane for establishing the physical connectivity. The overlay network is the logical topology that is built on top of the underlay network. The end hosts do not know about the overlay network. The overlay network uses encapsulation. For example, in GRE, it adds a GRE header on the IPv4 header.

As the fabric is built on top of a traditional network, it is sometimes referred to as the overlay network, and the traditional network is referred to as the underlay network.

Some common examples of overlay networks include GRE or mGRE, MPLS or VPLS, IPsec or DMVPN, CAPWAP, LISP, OTV, DFA, and ACI.

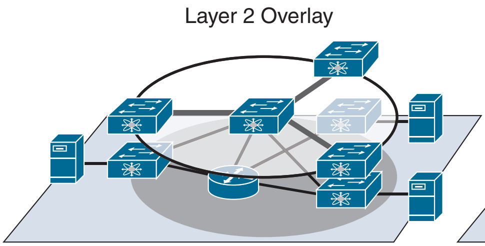
The underlay network can be used to establish physical connectivity using intelligent path control, load balancing, and high availability. The underlay network forms the simple forwarding plane.

The overlay network takes care of security, mobility, and programmability in the network. Using simple transport forwarding that provides redundant devices and paths, is simple and manageable, and provides optimized packet handling, the overlay network provides maximum reliability. Having a fabric in place enables several capabilities, such as the creation of virtual networks, user and device groups, and advanced reporting. Other capabilities include intelligent services for application recognition, traffic analytics, traffic prioritization, and traffic steering for optimum performance and operational effectiveness.

Fabric Overlay Types

There are generally two types of overlay fabric, as illustrated in Figure 18-4:

Figure 18-4 Layer 2 and Layer 3 Overlays



- **Layer 2 overlays:** Layer 2 overlays emulate a LAN segment and can be used to transport IP and non-IP frames. Layer 2 overlays carry a single subnet over the Layer 3 underlay. Layer 2 overlays are useful in emulating physical topologies and are subject to Layer 2 flooding.
- **Layer 3 overlays:** Layer 3 overlays abstract IP-based connectivity from physical connectivity and allow multiple IP networks as parts of each virtual network. Overlapping IP address space is supported across different Layer 3 overlays as long as the network virtualization is preserved outside the fabric, using existing network virtualization functions, such as VRF-Lite and MPLS L3VPN.

Fabric Underlay Provisioning

Fabric underlay provisioning can be done manually, or the process can be automated with Cisco DNA Center.

For an existing network where you have physical connectivity and routing configured, you can migrate to the Cisco SD-Access solution with a few primary considerations and requirements. First, there should be IP reachability within the network. The switches in the overlay are designated and configured as edge and border nodes. You must ensure that there is connectivity between the devices in the underlay network. Also, it is recommended to use IS-IS as the routing protocol. There are several advantages to using IS-IS, and it is easiest to automate the underlay with IS-IS as the routing protocol. IS-IS also has a few operational advantages, such as being able to neighbor up without IP address dependency. Also, the overlay network adds a fabric header to the IP header, so you need to consider the MTU in the network.

Underlay provisioning can be automated using Cisco DNA Center. The Cisco DNA Center LAN Automation feature is an alternative to manual underlay deployment for new networks and uses an IS-IS routed access design. Although there are many alternative routing protocols, IS-IS offers operational advantages such as neighbor establishment without IP protocol dependencies, peering capability using loopback addresses, and agnostic treatment of IPv4, IPv6, and non-IP traffic. The latest version of Cisco DNA Center LAN Automation uses Cisco Network Plug and Play features to deploy both unicast and multicast routing configuration in the underlay, aiding traffic delivery efficiency for services that are built on top.

Cisco SD-Access Fabric Data Plane and Control Plane

Cisco SD-Access configures the overlay network for fabric data plane encapsulation using the VXLAN technology framework. VXLAN encapsulates complete Layer 2 frames for transport across the underlay, with each overlay network identified by a VXLAN network identifier (VNI). The VXLAN header also carries the SGTs required for microsegmentation.

The function of mapping and resolving endpoint addresses requires a control plane protocol, and SD-Access uses Locator/ID Separation Protocol (LISP) for this task. LISP brings the advantage of routing based not only on the IP address or MAC address as the endpoint identifier (EID) for a device but also on an additional IP address that it provides as a routing locator (RLOC) to represent the network location of that device. The EID and RLOC combination provides all the necessary information for traffic forwarding, even if an endpoint uses an unchanged IP address when appearing in a different network location. Simultaneously, the decoupling of the endpoint identity from its location allows addresses in the same IP subnetwork to be available behind multiple Layer 3 gateways as opposed to the one-to-one coupling of IP subnetwork with the network gateway in traditional networks.

LISP and VXLAN are covered on Day 19, "LISP and VXLAN."

Cisco SD-Access Fabric Policy Plane

The Cisco SD-Access fabric policy plane is based on Cisco TrustSec. The VXLAN header carries the fields for virtual routing and forwarding (VRF) and Scalable Group Tags (SGTs) that are used in network segmentation and security policies.

Cisco TrustSec has a couple key features that are essential in the secure and scalable Cisco SD-Access solution. Traffic is segmented based on a classification group, called a scalable group, and not based on topology (VLAN or IP subnet). Based on endpoint classification, SGTs are assigned to enforce access policies for users, applications, and devices.

Cisco TrustSec provides software-defined segmentation that dynamically organizes endpoints into logical groups called security groups. Security groups, also known as scalable groups, are assigned based on business decisions using a richer context than an IP address. Unlike access control mechanisms that are based on network topology, Cisco TrustSec policies use logical groupings. Decoupling access entitlements from IP addresses and VLANs simplifies security policy maintenance tasks, lowers operational costs, and allows common access policies to be consistently applied to wired, wireless, and VPN access. By classifying traffic according to the contextual identity of the endpoint instead of its IP address, the Cisco TrustSec solution enables more flexible access controls for dynamic networking environments and data centers.

The ultimate goal of Cisco TrustSec technology is to assign a tag (SGT) to the user's or device's traffic at the ingress (inbound into the network) and then enforce the access policy based on the tag elsewhere in the infrastructure (for example, data center). Switches, routers, and firewalls use the SGT to make forwarding decisions. For instance, an SGT may be assigned to a Guest user so that the Guest traffic may be isolated from non-Guest traffic throughout the infrastructure.

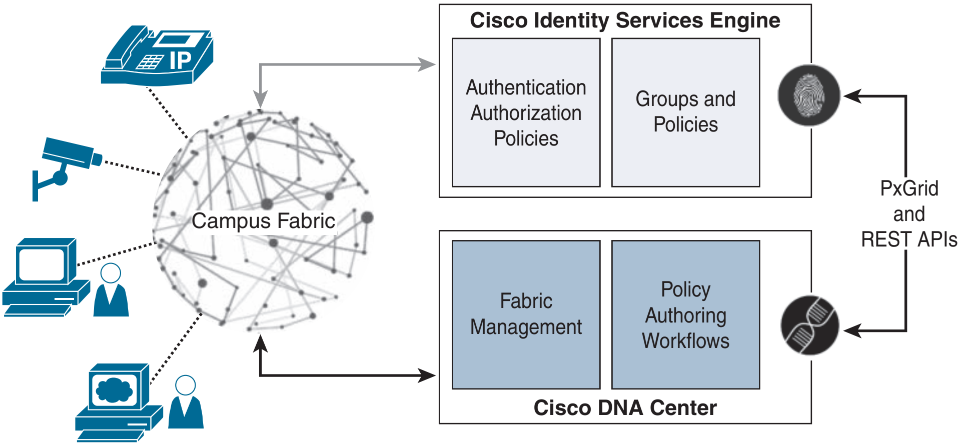
Note that the tags known in Cisco SD-Access as Scalable Group Tags (SGTs) were previously known as Security Group Tags in TrustSec and both terms reference the same segmentation tool.

Cisco TrustSec and ISE

Cisco Identity Services Engine (ISE) is a secure network access platform that enables increased management awareness, control, and consistency for users and devices accessing an organization's network. ISE is a part of Cisco SD-Access for policy implementation, enabling dynamic mapping of users and devices to scalable groups and simplifying end-to-end security policy enforcement.

In ISE, users and devices are shown in a simple and flexible interface. ISE integrates with Cisco DNA Center by using Cisco Platform Exchange Grid (pxGrid) and REST APIs for exchange of client information and automation of fabric-related configurations on ISE. The Cisco SD-Access solution integrates Cisco TrustSec by supporting group-based policy end-to-end, including SGT information in the VXLAN headers for data plane traffic while supporting multiple VNs using unique VNI assignments. Figure 18-5 illustrates the relationship between ISE and Cisco DNA Center.

Figure 18-5 Cisco ISE and Cisco DNA Center

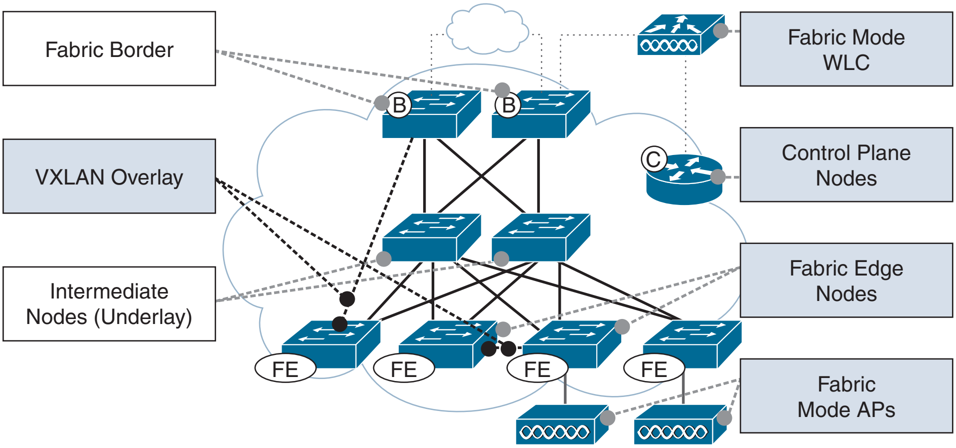

IP
Campus Fabric
Cisco Identity Services Engine
Authentication Authorization Policies
Groups and Policies
PxGrid and REST APIs
Fabric Management
Policy Authoring Workflows
Cisco DNA Center

Authentication, authorization, and accounting (AAA) services, groups, policy, and endpoint profiling are driven by ISE and orchestrated by Cisco DNA Center's policy authoring workflows. A scalable group is identified by the SGT, a 16-bit value that is transmitted in the VXLAN header. SGTs are centrally defined, managed, and administered by Cisco ISE. ISE and Cisco DNA Center are tightly integrated through REST APIs, and management of the policies is driven by Cisco DNA Center. ISE supports standalone and distributed deployment models. Also, multiple distributed nodes can be deployed together, which supports failover resiliency. The range of options enables support for hundreds of thousands of endpoint devices, and a subset of the devices are used for Cisco SD-Access. At minimum, a basic two-node ISE deployment is recommended for Cisco SD-Access deployments, with each node running all services for redundancy. Cisco SD-Access fabric edge node switches send authentication requests to the Policy Services Node (PSN) persona running on ISE. In the case of a standalone deployment, with or without node redundancy, that PSN persona is referenced by a single IP address. An ISE distributed model uses multiple active PSN personas, each with a unique address. All PSN addresses are learned by Cisco DNA Center, and the Cisco DNA Center user maps fabric edge node switches to the PSN that supports each edge node.

Cisco SD-Access Fabric Components

The campus fabric is composed of fabric control plane nodes, edge nodes, intermediate nodes, and border nodes. Figure 18-6 illustrates the entire Cisco SD-Access solution and its components.

Figure 18-6 Cisco SD-Access Solution and Fabric Components



Fabric devices have different functionality, depending on their roles. These are the basic roles of the devices:

- **Control plane node:** LISP map server/map resolver (MS/MR) that manages EID to device relationships.
- **Border node:** A fabric device (such as a core layer device) that connects external Layer 3 networks to the Cisco SD-Access fabric.
- **Edge node:** A fabric device (such as an access or distribution layer device) that connects wired endpoints to the Cisco SD-Access fabric.
- **Fabric wireless controller:** A WLC that is fabric enabled.
- **Fabric mode AP:** An access point that is fabric enabled.
- **Intermediate node:** An underlay device.

These fabric nodes are explained in more detail in the following sections.

Cisco SD-Access Control Plane Node

The Cisco SD-Access fabric control plane node is based on the LISP map server (MS) and map resolver (MR) functionality combined on the same node. The control plane database tracks all endpoints in the fabric site and associates the endpoints to fabric nodes, decoupling the endpoint IP address or MAC address from the location (closest router) in the network. The control plane node functionality can be collocated with a border node or can use dedicated nodes for scale; between two and six nodes are used for resiliency. Border and edge nodes register with and use all control plane nodes, so the resilient nodes chosen should be of the same type for consistent performance.

Cisco SD-Access Edge Node

The Cisco SD-Access fabric edge node is the equivalent of an access layer switch in a traditional campus LAN design. Edge nodes implement a Layer 3 access design with the addition of the following fabric functions:

- **Endpoint registration:** An edge node informs the control plane node when an endpoint is detected.
- **Mapping of user to virtual network:** An edge node assigns a user to an SGT for segmentation and policy enforcement.
- **Anycast Layer 3 gateway:** One common gateway is used for all nodes in a shared EID subnet.
- **LISP forwarding:** Fabric edge nodes query the map resolver to determine the RLOC associated with the destination EID and use that information as the traffic destination.
- **VXLAN encapsulation/decapsulation:** Fabric edge nodes use the RLOC associated with the destination IP address to encapsulate the traffic with VXLAN headers. Similarly, VXLAN traffic received at a destination RLOC is decapsulated.

Cisco SD-Access Border Node

A fabric border node serves as a gateway between the Cisco SD-Access fabric site and the networks external to the fabric. A fabric border node is responsible for network virtualization interworking and SGT propagation from the fabric to the rest of the network. A fabric border node can be configured as an internal border, operating as the gateway for specific network addresses, such as a shared services or data center network, or as an external border, useful as a common exit point from a fabric, such as for the rest of an enterprise network and the Internet. A border node can also have a combined role as an anywhere border (both internal and external border).

Border nodes implement the following functions:

- **Advertisement of EID subnets:** Cisco SD-Access configures Border Gateway Protocol (BGP) as the preferred routing protocol used to advertise the EID prefixes outside the fabric, and traffic destined to EID subnets from outside the fabric goes through the border nodes.
- **Fabric domain exit point:** The external fabric border is the gateway of last resort for the fabric edge nodes.
- **Mapping of LISP instance to VRF instance:** The fabric border can extend network virtualization from inside the fabric to outside the fabric by using external VRF instances to preserve the virtualization.
- **Policy mapping:** The fabric border node maps SGT information from within the fabric to be appropriately maintained when exiting that fabric.

Cisco SD-Access Intermediate Node

The fabric intermediate nodes are part of the Layer 3 network that interconnects the edge nodes to the border nodes. In a three-tier campus design using core, distribution, and access layers, the fabric intermediate nodes are equivalent to distribution switches. Fabric intermediate nodes only route the IP traffic inside the fabric. No VXLAN encapsulation and decapsulation or LISP control plane messages are required from the fabric intermediate node.

Cisco SD-Access Wireless LAN Controller and Fabric Mode Access Points (APs)

Fabric Wireless LAN Controller

The fabric WLC integrates with the control plane for wireless and the fabric control plane. Both fabric WLCs and non-fabric WLCs provide AP image and configuration management, client session management, and mobility services. Fabric WLCs provide additional services for fabric integration by registering MAC addresses of wireless clients into the host-tracking database of the fabric control plane during wireless client join events and by supplying fabric edge RLOC location updates during client roaming events.

A key difference with non-fabric WLC behavior is that fabric WLCs are not active participants in the data plane traffic-forwarding role for the SSIDs that are fabric enabled; fabric mode APs directly forward traffic through the fabric for those SSIDs.

Typically, the fabric WLC devices connect to a shared services distribution or data center outside the fabric and fabric border, which means their management IP address exists in the global routing table. For the wireless APs to establish a Control and Provisioning of Wireless Access Points (CAPWAP) tunnel for WLC management, the APs must be in a virtual network that has access to the external device. In the Cisco SD-Access solution, Cisco DNA Center configures wireless APs to reside within the VRF instance named INFRA_VRF, which maps to the global routing table; this eliminates the need for route leaking or fusion router (multi-VRF router selectively sharing routing information) services to establish connectivity.

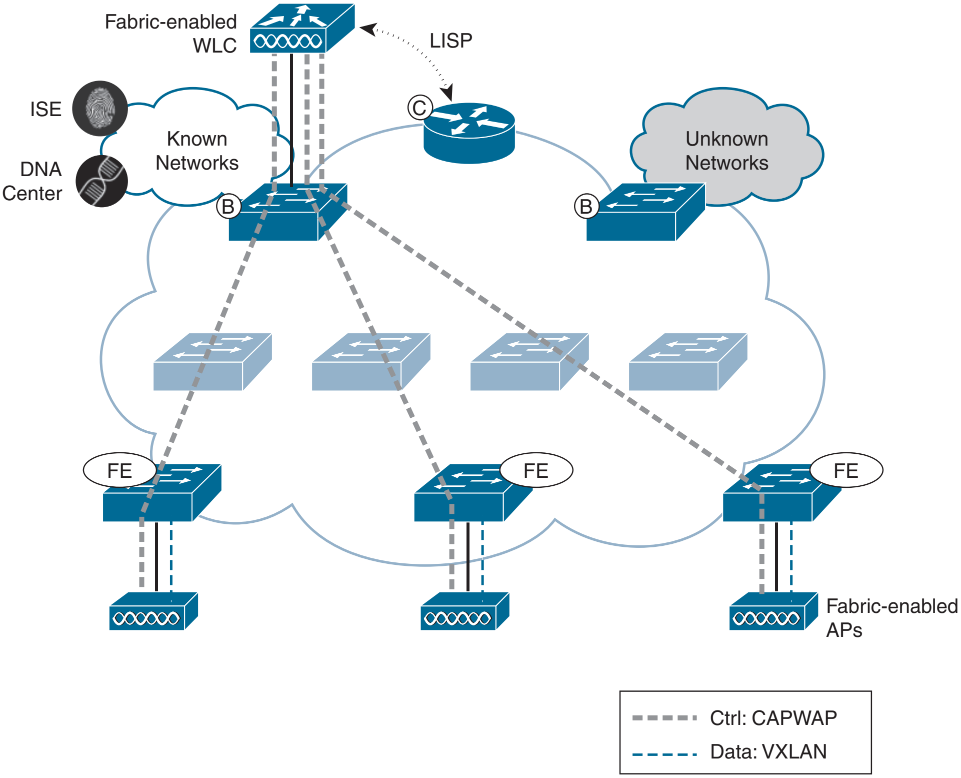
Fabric Mode Access Points

The fabric mode APs are Cisco Wifi6 (802.1ax) and Cisco 802.11ac Wave 2 and Wave 1 APs associated with the fabric WLC that have been configured with one or more fabric-enabled SSIDs. Fabric mode APs continue to support the same 802.11ac wireless media services that traditional APs support; support Cisco Application Visibility and Control (AVC), quality of service (QoS), and other wireless policies; and establish the CAPWAP control plane to the fabric WLC. Fabric APs join as local mode APs and must be directly connected to the fabric edge node switch to enable fabric registration events, including RLOC assignment via the fabric WLC. The APs are recognized by the fabric edge nodes as special wired hosts and assigned to a unique overlay network in a common EID space across a fabric. The assignment allows management simplification by using a single subnet to cover the AP infrastructure at a fabric site.

When wireless clients connect to a fabric mode AP and authenticate into the fabric-enabled wireless LAN, the WLC updates the fabric mode AP with the client Layer 2 VNI and an SGT supplied by ISE. Then the WLC registers the wireless client Layer 2 EID into the control plane, acting as a proxy for the egress fabric edge node switch. After the initial connectivity is established, the AP uses the Layer 2 VNI information to VXLAN encapsulate wireless client communication on the Ethernet connection to the directly connected fabric edge switch. The fabric edge switch maps the client traffic to the appropriate VLAN interface associated with the VNI for forwarding across the fabric and registers the wireless client IP addresses with the control plane database.

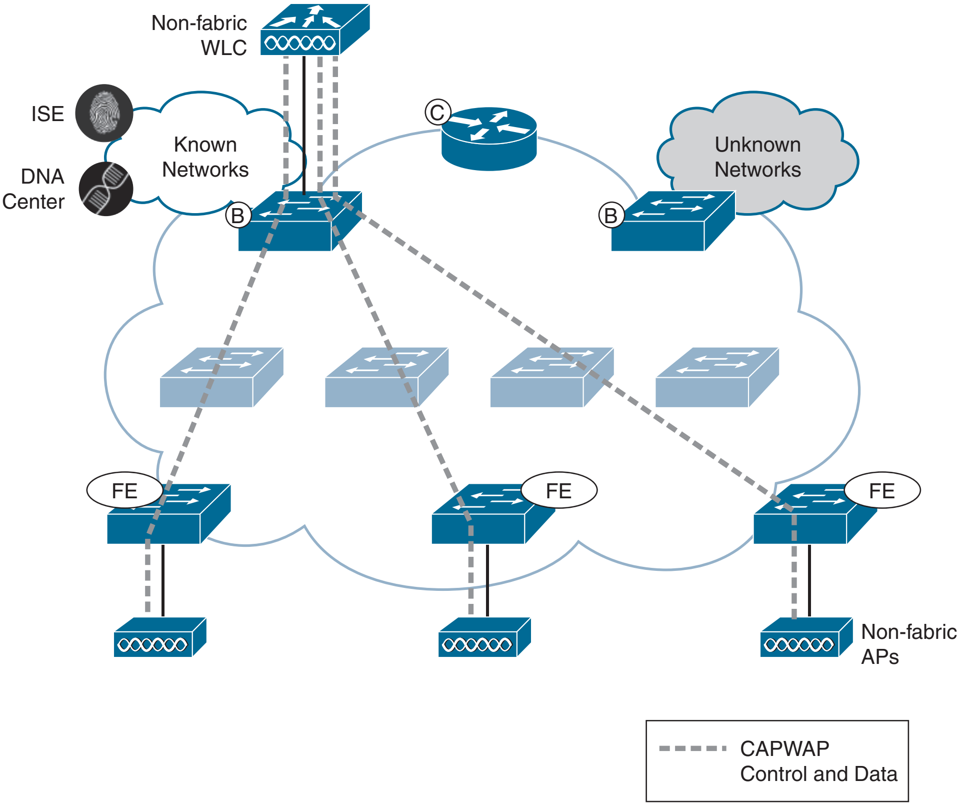
Figure 18-7 illustrates how fabric-enabled APs establish a CAPWAP tunnel with the fabric-enabled WLC for control plane communication, but the same APs use VXLAN to tunnel traffic directly within the Cisco SD-Access fabric. This is an improvement over the traditional Cisco Unified Wireless Network (CUWN) design, which requires all wireless traffic to be tunneled to the WLC.

Figure 18-7 Cisco SD-Access Wireless Traffic Flow



If the network needs to support older model APs, it is possible to also use the over-the-top method of wireless integration with the SD-Access fabric. When you use this method, the control plane and data plane traffic from the APs continues to use CAPWAP-based tunnels. In this mode, the Cisco SD-Access fabric provides only a transport to the WLC. This method can also be used as a migration step to full Cisco SD-Access in the future. Figure 18-8 illustrates this type of solution, where control and data traffic is tunneled from the APs to the WLC. Notice the lack of LISP control plane connection between the WLC and the fabric control plane node.

Figure 18-8 Cisco CUWN Wireless Over the Top


Non-fabric WLC
ISE
DNA Center
Known Networks
C
B
Unknown Networks
B
FE
FE
FE
Non-fabric APs
CAPWAP Control and Data

Shared Services in Cisco SD-Access

Designing for end-to-end network virtualization requires detailed planning to ensure the integrity of the virtual networks. In most cases, there is a need to have some form of shared services that can be reused across multiple virtual networks. It is important that those shared services be deployed correctly to preserve the isolation between different virtual networks sharing those services. The use of a fusion router directly attached to the fabric border provides a mechanism for route leaking of shared services prefixes across multiple networks, and the use of firewalls provides an additional layer of security and monitoring of traffic between virtual networks. Examples of shared services that exist outside the Cisco SD-Access fabric include the following:

- DHCP, DNS, and IP address management
- Internet access
- Identity services (such as AAA/RADIUS)
- Data collectors (NetFlow and syslog)
- Monitoring (SNMP)

- Time synchronization (NTP)
- IP voice/video collaboration services

Fusion Router

The generic term *fusion router* comes from the MPLS Layer 3 VPN. The basic idea is that a fusion router is aware of the prefixes available inside each VPN instance, either because of static routing configuration or through route peering, and it therefore is able to fuse these routes together. A generic fusion router's responsibilities are to route traffic between separate VRF instances (VRF leaking) or to route traffic to and from a VRF instance to a shared pool of resources such as DHCP and DNS servers in the global routing table (route leaking in the GRT). Both responsibilities involve moving routes from one routing table into a separate VRF routing table.

In a Cisco SD-Access deployment, the fusion router has a single responsibility: to provide access to shared services for the endpoints in the fabric. There are two primary ways to accomplish this task, depending on how the shared services are deployed. The first option is used when the shared services routes are in the GRT. On the fusion router, IP prefix lists are used to match the shared services routes, route maps reference the IP prefix lists, and the VRF configurations reference the route maps to ensure that only the specifically matched routes are leaked. The second option is to place shared services in a dedicated VRF instance on the fusion router. With shared services in a VRF instance and the fabric endpoints in other VRF instances, route targets are used to leak traffic between instances.

A fusion router can be either a true routing platform, a Layer 3 switching platform, or a firewall that must meet several technological requirements to support VRF routing.

Figure 18-9 illustrates the use of a fusion router. In this example, the services infrastructure is placed into a dedicated VRF context of its own, and VRF route leaking needs to be provided in order for the virtual network (VRF instance) in Cisco SD-Access fabric to have continuity of connectivity to the services infrastructure. The methodology used to achieve continuity of connectivity in the fabric for the users is to deploy a fusion router connected to the Cisco SD-Access border through VRF-Lite using BGP/IGP, and the services infrastructure is connected to the fusion router in a services VRF instance.

Figure 18-9 Cisco SD-Access Fusion Router Role

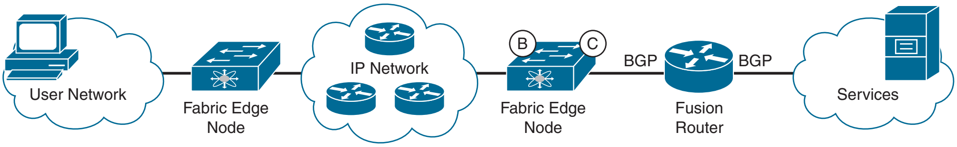
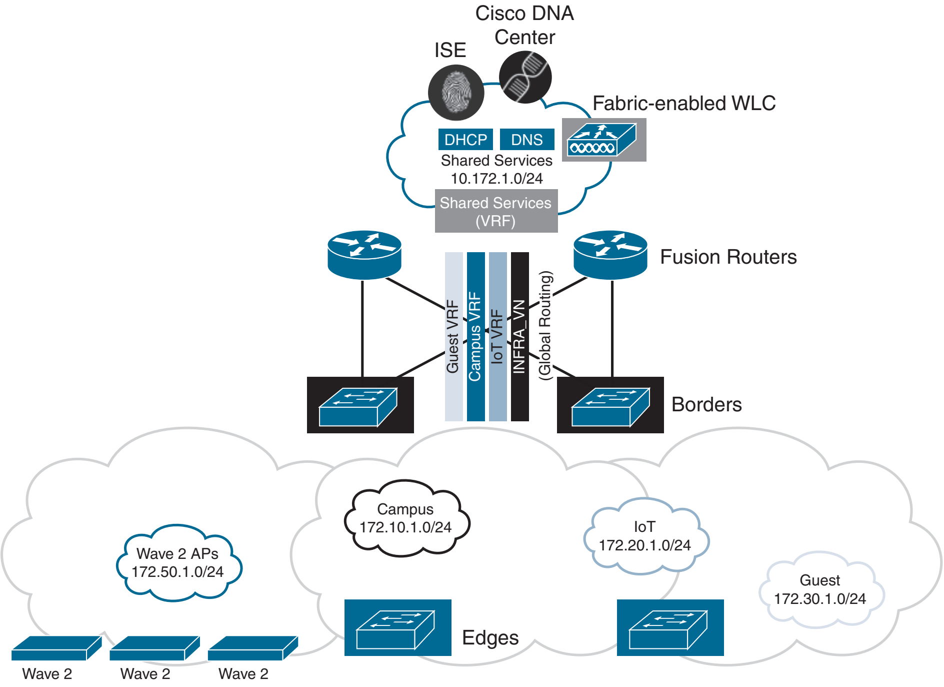


Figure 18-10 illustrates a complete Cisco SD-Access logical topology that uses three VRF instances within the fabric (Guest, Campus, and IoT), as well as a shared services VRF instance that the fusion router leaks into the other VRF instances. The WLC and APs are all fabric-enabled devices in this example. The INFRA_VN VRF instance is used for APs and extended nodes, and its VRF/VN is leaked to the global routing table (GRT) on the borders. INFRA_VN is used for the Plug and Play (PnP) onboarding services for these devices through Cisco DNA Center. Note that INFRA_VN cannot be used for other endpoints and users.

Figure 18-10 Cisco SD-Access Logical Topology


Cisco DNA Center
ISE
Fabric-enabled WLC
DHCP
DNS
Shared Services
10.172.1.0/24
Shared Services
(VRF)
Fusion Routers
Guest VRF
Campus VRF
IoT VRF
INFRA_VN
(Global Routing)
Borders
Campus
172.10.1.0/24
IoT
172.20.1.0/24
Wave 2 APs
172.50.1.0/24
Guest
172.30.1.0/24
Edges
Wave 2
Wave 2
Wave 2

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	23
CCNP Enterprise Design ENSLD 300-420 Official Cert Guide: Designing Cisco Enterprise Networks	10
Cisco Software-Defined Access Design Guide	https://cs.co/sda-sdg

This page intentionally left blank

Day 17

SD-WAN

ENCOR 350-401 Exam Topics

Architecture

- Explain the working principles of the Cisco SD-WAN solution
  - SD-WAN control and data planes elements
  - Traditional WAN and SD-WAN solutions

Key Topics

Today we review the Cisco SDN technology Cisco Software-Defined WAN (SD-WAN). Cisco SD-WAN is an enterprise-grade WAN architecture overlay that enables digital and cloud transformation for enterprises. It fully integrates routing, security, centralized policy, and orchestration into large-scale networks. It is multitenant, cloud-delivered, highly automated, secure, scalable, and application-aware, with rich analytics. Recall that SDN is a centralized approach to network management that abstracts away the underlying network infrastructure from its applications. This decoupling of the data plane from the control plane allows you to centralize the intelligence of the network and makes possible more network automation, simplification of operations, and centralized provisioning, monitoring, and troubleshooting. Cisco SD-WAN applies these principles of SDN to the WAN. The focus today is on the Cisco SD-WAN enterprise solution based on technology acquired from Viptela.

Software-Defined WAN

New technologies have been developed to handle the growing demand that new applications, devices, and services are placing on the enterprise WAN. This section describes the need for Cisco SD-WAN and the major components of SD-WAN and its basic operations. The Cisco SD-WAN technology addresses problems and challenges of common WAN deployments, such as the following:

- Centralized network and policy management, as well as operational simplicity, resulting in reduced change control and deployment times.
- A mix of MPLS and low-cost broadband or any combination of transports in an active/active fashion, optimizing capacity and reducing bandwidth costs.

- A transport-independent overlay that extends to the data center, branch, and cloud.
- Deployment flexibility. Due to the separation of the control plane and data plane, controllers can be deployed on premises or in the cloud or a combination of both. Cisco SD-WAN Edge router deployment can be physical or virtual; these routers can be deployed anywhere in the network.
- Robust and comprehensive security, which includes strong encryption of data, end-to-end network segmentation, router and controller certificate identity with a zero-trust security model, control plane protection, application firewall, and insertion of Cisco Umbrella, firewalls, and other network services.
- Seamless connectivity to the public cloud and movement of the WAN edge to the branch.
- Application visibility and recognition in addition to application-aware policies with real-time service-level agreement (SLA) enforcement.
- Dynamic optimization of software-as-a-service (SaaS) applications, resulting in improved application performance for users.
- Rich analytics with visibility into applications and infrastructure, enabling rapid troubleshooting and assisting in forecasting and analysis for effective resource planning.

Need for Cisco SD-WAN

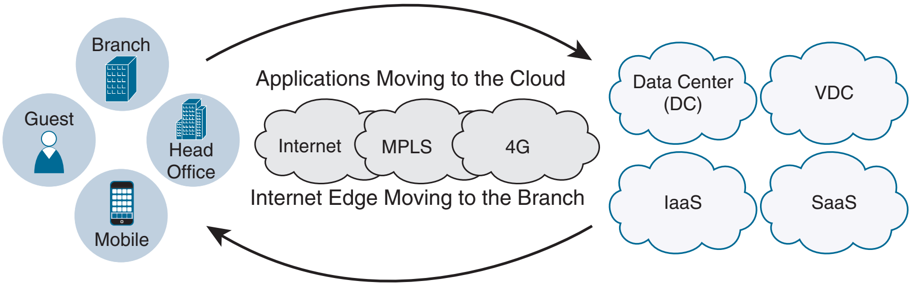
Applications used by enterprise organizations have evolved over the past several years. As a result, enterprise WANs are evolving to handle the rapidly changing needs of these newer, higher-resource-consuming applications.

Wide-area networking is evolving to manage a changing application landscape that has a greater demand for mobile and Internet of Things (IoT) device traffic, SaaS applications, infrastructure as a service (IaaS), and cloud adoption. In addition, security requirements are increasing, and applications are requiring prioritization and optimization.

Legacy WAN architectures are facing major challenges in this evolving landscape. Legacy WAN architectures typically consist of multiple Multiprotocol Label Switching (MPLS) transports or an MPLS transport paired with an Internet or 4G/5G/LTE (Long-Term Evolution) transport used in an active and backup fashion, most often with Internet or SaaS traffic being backhauled to a central data center or regional hub. Issues with these architectures include insufficient bandwidth, high bandwidth costs, application downtime, poor SaaS performance, complex operations, complex workflows for cloud connectivity, long deployment times and policy changes, limited application visibility, and difficulty securing the network.

Figure 17-1 illustrates the transition that is occurring in WANs today with applications moving to the cloud and the Internet edge moving to the branch office.

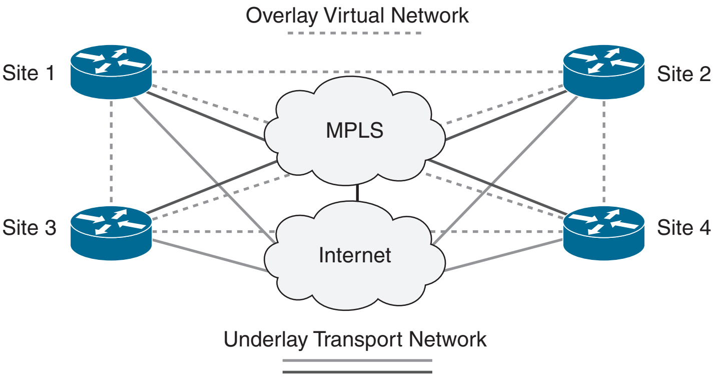
Figure 17-1 Need for Cisco SD-WAN


Branch
Guest
Head Office
Mobile
Applications Moving to the Cloud
Internet
MPLS
4G
Internet Edge Moving to the Branch
Data Center (DC)
VDC
IaaS
SaaS

Cisco SD-WAN represents a shift from the older, hardware-based legacy WAN model to a secure, software-based virtual IP fabric overlay that runs over standard network transport services.

The Cisco SD-WAN solution is a software-based virtual IP fabric overlay network that builds secure, unified connectivity over any transport network (the underlay). The underlay transport network, which is the physical infrastructure for the WAN, may be the public internet, MPLS, Metro Ethernet, or LTE/4G/5G (when available). The underlay network provides a service to the overlay network and is responsible for the delivery of packets across networks. Figure 17-2 illustrates the relationship between underlay and overlay networks in the Cisco SD-WAN solution.

Figure 17-2 Cisco SD-WAN Underlay and Overlay Networks


Overlay Virtual Network
Site 1
Site 2
MPLS
Site 3
Site 4
Internet
Underlay Transport Network

SD-WAN Architecture and Components

Cisco SD-WAN is based on the same routing principles that have been used on the Internet for years. The Cisco SD-WAN separates the data plane from the control plane and virtualizes much of the routing that used to require dedicated hardware. True separation between the control and data plane enables the Cisco SD-WAN solution to run over any transport circuits.

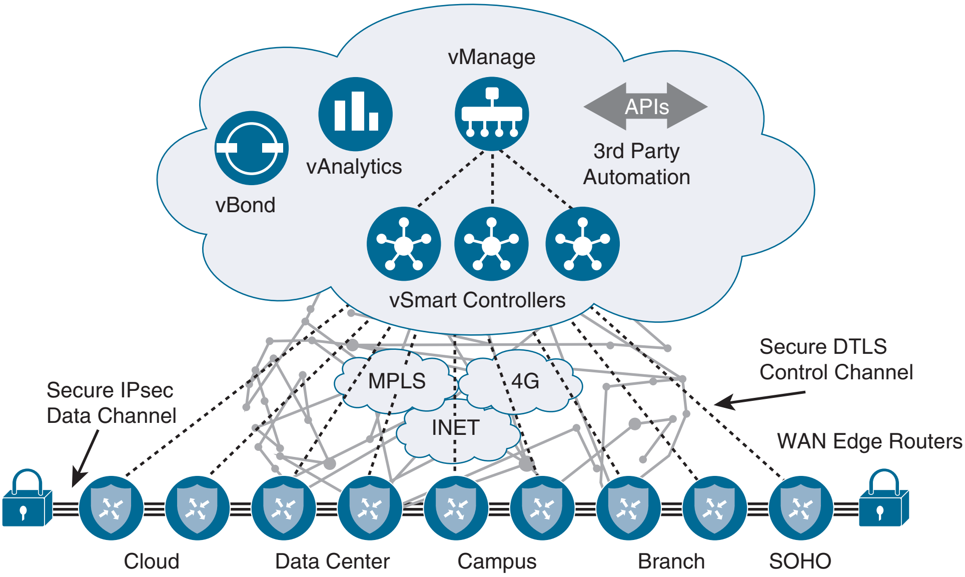
The virtualized network runs as an overlay on cost-effective hardware, such as physical routers, called WAN Edge routers, and virtual machines (VMs) in the cloud, called WAN Edge cloud routers. Centralized controllers, called vSmart controllers, oversee the control plane of the SD-WAN fabric, efficiently managing provisioning, maintenance, and security for the entire Cisco SD-WAN overlay network. The vBond orchestrator automatically authenticates all other SD-WAN devices when they join the SD-WAN overlay network.

The control plane manages the rules for routing traffic through the overlay network, and the data plane passes the actual data packets among the network devices. The control plane and data plane form the fabric for each customer's deployment, according to customer requirements, over existing circuits.

The vManage network management system (NMS) provides a simple yet powerful set of graphical dashboards for monitoring network performance on all devices in the overlay network from a centralized monitoring station. In addition, the vManage NMS provides centralized software installation, upgrades, and provisioning, whether for a single device or as a bulk operation for many devices simultaneously.

Figure 17-3 provides an overview of the Cisco SD-WAN architecture and its components.

Figure 17-3 Cisco SD-WAN Solution Architecture


vManage
APIs
3rd Party Automation
vAnalytics
vBond
vSmart Controllers
MPLS
4G
INET
Secure IPsec Data Channel
Secure DTLS Control Channel
WAN Edge Routers
Cloud
Data Center
Campus
Branch
SOHO

SD-WAN Orchestration Plane

The Cisco vBond orchestrator is a multitenant element of the Cisco SD-WAN fabric. vBond is the first point of contact and performs initial authentication when devices are connecting to the organization overlay. vBond facilitates the mutual discovery of the control and management elements of the fabric by using a zero-trust certificate-based allowed-list model. Cisco vBond automatically distributes a list of vSmart controllers and the vManage system to the WAN Edge routers during the deployment process.

For situations in which vSmart controllers, the vManage system, or the WAN Edge routers are behind NAT, the vBond orchestrator facilitates the function of NAT traversal by allowing the learning of public (post-NAT) and private (pre-NAT) IP addresses. The discovery of public and private

IP addresses allows connectivity to be established across public (Internet, 4G/5G/LTE) and private (MPLS, point-to-point) WAN transports.

The vBond orchestrator should reside in the public IP space or on the private IP space with 1:1 NAT, so that all remote sites, especially Internet-only sites, can reach it. When tied to DNS, this reachable vBond IP address allows for a zero-touch deployment.

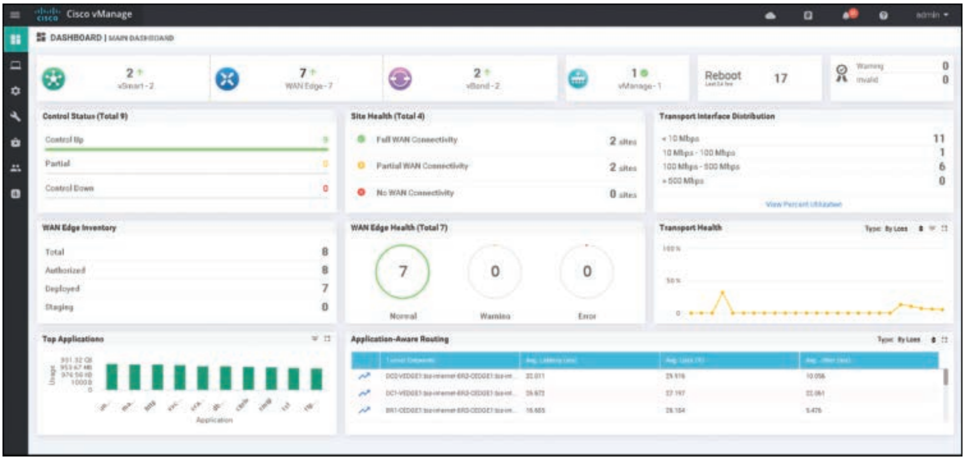
vBond should be highly resilient. If vBond is down, no other device can join the overlay. When vBond is deployed as an on-premises solution by the customer, it is the responsibility of the customer to provide adequate infrastructure resiliency with multiple vBond orchestrators. Another solution is for the vBond orchestrator to be cloud hosted with Cisco SD-WAN CloudOps. With Cisco CloudOps, Cisco deploys the Cisco SD-WAN controllers—specifically the Cisco vManage NMS, the Cisco vBond orchestrator, and the Cisco vSmart controller—on the public cloud. Cisco then provides administrator access. By default, a single Cisco vManage NMS, Cisco vBond orchestrator, and Cisco vSmart controller are deployed in the primary cloud region, and an additional Cisco vBond orchestrator and Cisco vSmart controller are deployed in the secondary, or backup, region.

SD-WAN Management Plane

Cisco vManage is on the management plane and provides a single pane of glass for Day 0, Day 1, and Day 2 operations. Cisco vManage's multitenant web-scale architecture meets the needs of enterprises and service providers alike.

Cisco vManage has a web-based GUI with role-based access control (RBAC). Some key functions of Cisco vManage include centralized provisioning, centralized policies and device configuration templates, and the ability to troubleshoot and monitor the entire environment. You can also perform centralized software upgrades on all fabric elements, including WAN Edge, vBond, vSmart, and vManage. The vManage GUI is illustrated in Figure 17-4.

Figure 17-4 Cisco SD-WAN vManage GUI


Cisco vManage
admin
DASHBOARD | MAIN DASHBOARD
2 vSmart - 2
7 WAN Edge - 7
2 vBond - 2
1 vManage - 1
Reboot 17
Warning 0
Invalid 0
Control Status (Total 9)
Control Up 9
Partial 0
Control Down 0
Site Health (Total 4)
Full WAN Connectivity 2 sites
Partial WAN Connectivity 2 sites
No WAN Connectivity 0 sites
Transport Interface Distribution
< 10 Mbps 11
10 Mbps - 100 Mbps 1
100 Mbps - 500 Mbps 6
> 500 Mbps 0
View Percent Utilization
WAN Edge Inventory
Total 8
Authorized 8
Deployed 7
Staging 0
WAN Edge Health (Total 7)
7 Normal
0 Warning
0 Error
Transport Health
Type: By Loss
100 %
50 %
0
Top Applications
Usage
Application
Application-Aware Routing
Type: By Loss

vManage should run in high resiliency mode because if you lose vManage, you lose the management plane.

vManage supports multitenant mode in addition to the default single-tenant mode of operation.

You can use vManage's programmatic interfaces to enable DevOps operations and to also extract performance statistics collected from the entire fabric. You can export performance statistics to external systems or to the Cisco vAnalytics tool for further processing and closer examination.

Cisco SD-WAN software provides a REST API, which is a programmatic interface for controlling, configuring, and monitoring the Cisco SD-WAN devices in an overlay network. You access the REST API through the vManage web server.

A REST API is a web service API that adheres to the REST (Representational State Transfer) architecture. The REST architecture uses a stateless, client/server, cacheable communications protocol. The vManage NMS web server uses HTTP or its secure counterpart, HTTPS, as the communications protocol. REST applications communicate over HTTP or HTTPS by using standard HTTP methods to make calls between network devices.

REST is a simpler alternative to mechanisms such as remote procedure calls (RPCs) and web services such as Simple Object Access Protocol (SOAP) and Web Service Definition Language (WSDL).

SD-WAN Control Plane

The control plane is the centralized brain of the solution, establishing Overlay Management Protocol (OMP) peering with all the WAN Edge routers. Control plane policies such as service chaining, traffic engineering, and per-VPN topology are implemented by the control plane. The goal of the control plane is to dramatically reduce complexity in the entire fabric network. While no network data is forwarded by the control plane itself, connectivity information is distributed from the control plane to all WAN Edge routers to orchestrate the secure data plane of the fabric.

Cisco vSmart controllers provide scalability to the control plane functionality of the Cisco SD-WAN fabric. The vSmart controllers facilitate fabric discovery by running OMP between themselves and the WAN Edge routers. The vSmart controller acts as a distribution point to establish data plane connectivity between the WAN Edge routers. This information exchange includes service LAN-side reachability, transport WAN-side IP addressing, IPsec encryption keys, site identifiers, and so on. Together with WAN Edge routers, vSmart controllers act as a distribution system for the pertinent information required to establish data plane connectivity directly between the WAN Edge routers.

All control plane updates are sent from WAN Edge to vSmart in a route-reflector fashion. vSmart then reflects those updates to all remote WAN Edge sites. This is how every WAN Edge learns about all the available tunnel endpoints and user prefixes in the network. Because the control plane is centralized, you are not required to build control channels directly between all WAN Edge routers. vSmart controllers also distribute data plane and application-aware routing policies to the WAN Edge routers for enforcement. Control policies, acting on the control plane information, are locally enforced on the vSmart controllers. These control plane policies can implement service chaining and various types of topologies, and they generally can influence the flow of traffic across the fabric.

The use of a centralized control plane dramatically reduces the control plane load traditionally associated with building large-scale IPsec networks, solving the *n*^2 complexity problem. The vSmart controller deployment model solves the horizontal scale issue and also provides high availability and resiliency. vSmart controllers are often deployed in geographically dispersed data centers to reduce the likelihood of control plane failure. When delivered as a cloud service, vSmart controllers are redundantly hosted by Cisco CloudOps. When deployed as an on-premises solution by the customer, the customer must provide infrastructure resiliency.

SD-WAN Data Plane

A WAN Edge router functions as the data plane. The WAN Edge routers provide a secure data plane with remote WAN Edge routers and a secure control plane with vSmart controllers, and they implement data plane and application-aware policies. Because all data within the fabric is forwarded in the data plane, performance statistics are exported from the WAN Edge routers.

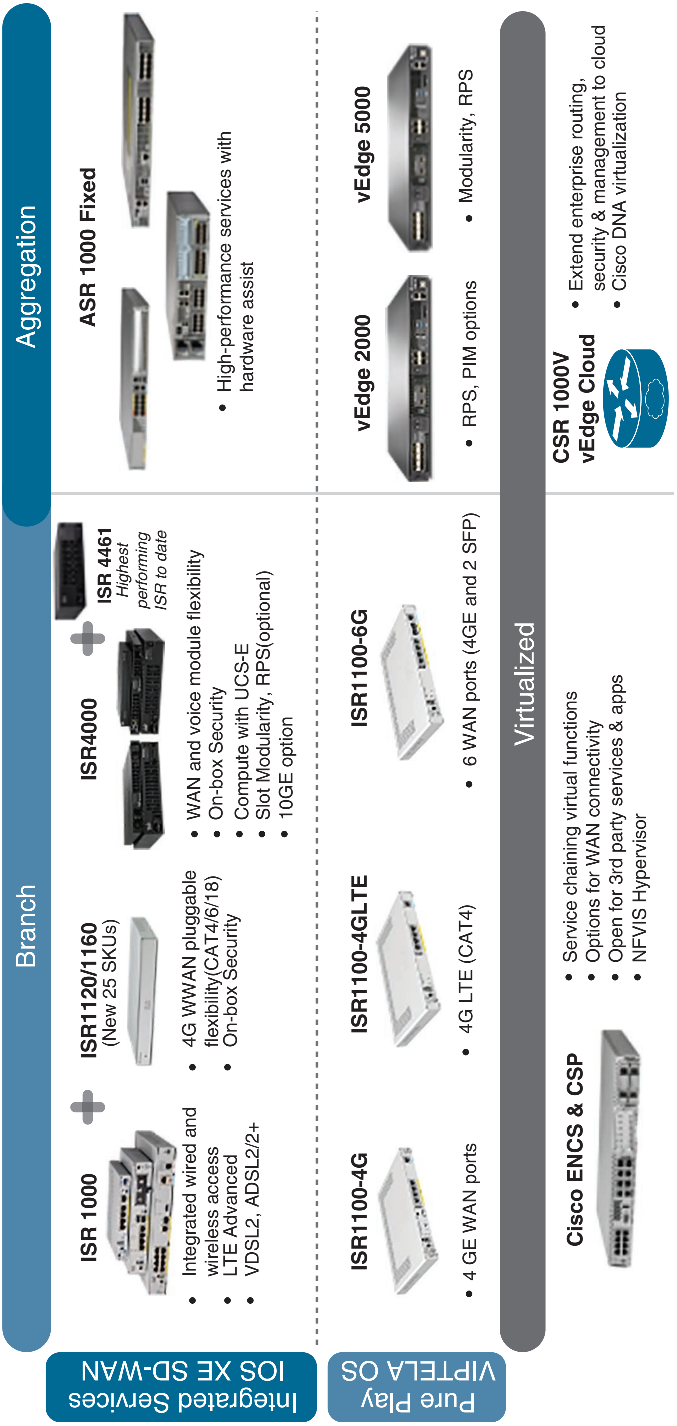
Cisco WAN Edge routers are positioned at every site at which the Cisco SD-WAN fabric must be extended. WAN Edge routers are responsible for encrypting and decrypting application traffic between the sites. The WAN Edge routers establish a control plane relationship with the vSmart controller to exchange the pertinent information required to establish the fabric and learn centrally provisioned policies. Data plane and application-aware routing policies are implemented on the WAN Edge routers. WAN Edge routers export performance statistics and alerts and events to the centralized vManage system for a single point of management.

WAN Edge routers use standards-based OSPF and BGP routing protocols for learning reachability information from service LAN-side interfaces and for brownfield integration with non-SD-WAN sites. WAN Edge routers have a very mature full-stack routing implementation, which accommodates simple, moderate, and complex routed environments. For Layer 2 redundant service LAN-side interfaces, WAN Edge routers implement the Virtual Router Redundancy Protocol (VRRP) first-hop redundancy protocol, which can operate on a per-VLAN basis. WAN Edge routers can be brought online in a full zero-touch deployment fashion or by requiring administrative approval. Zero-touch deployment relies on the use of signed certificates installed in the onboard Trusted Platform Module (TPM) to establish a unique router identity.

Finally, WAN Edge routers are available in both physical and virtual form factors. Physical form factors are deployed as appliances offering 100 Mbps, 1 Gbps, or 10 Gbps, depending on the throughput needs. The virtual form factor can be deployed in public clouds, such as AWS and Microsoft Azure, or as network functions virtualization (NFV) on the virtual customer-premises equipment/universal customer-premises equipment (vCPE/uCPE) platforms with the use of kernel-based virtual machine (KVM) or Elastic Sky X Integrated (ESXi) hypervisors.

Note that there are also two general types of WAN Edge routers: the original Viptela platform routers running Viptela software (vEdge) and the Cisco IOS XE routers running SD-WAN code (cEdge). Figure 17-5 shows the different platform options available for deploying Cisco SD-WAN WAN Edge devices.

Figure 17-5 Cisco SD-WAN Platform Options


Integrated Services
IOS XE SD-WAN
Pure Play
VIPTELA OS
Branch
Aggregation
ISR 1000
• Integrated wired and wireless access
• LTE Advanced
• VDSL2, ADSL2/2+
ISR1120/1160
(New 25 SKUs)
• 4G WWAN pluggable flexibility(CAT4/6/18)
• On-box Security
ISR4000
ISR 4461
Highest performing ISR to date
• WAN and voice module flexibility
• On-box Security
• Compute with UCS-E
• Slot Modularity, RPS(optional)
• 10GE option
ASR 1000 Fixed
• High-performance services with hardware assist
ISR1100-4G
• 4 GE WAN ports
ISR1100-4GLTE
• 4G LTE (CAT4)
ISR1100-6G
• 6 WAN ports (4GE and 2 SFP)
vEdge 2000
• RPS, PIM options
vEdge 5000
• Modularity, RPS
Virtualized
Cisco ENCS & CSP
• Service chaining virtual functions
• Options for WAN connectivity
• Open for 3rd party services & apps
• NFVIS Hypervisor
CSR 1000V
vEdge Cloud
• Extend enterprise routing, security & management to cloud
• Cisco DNA virtualization

@cisco_in_persian_channel

SD-WAN Automation and Analytics

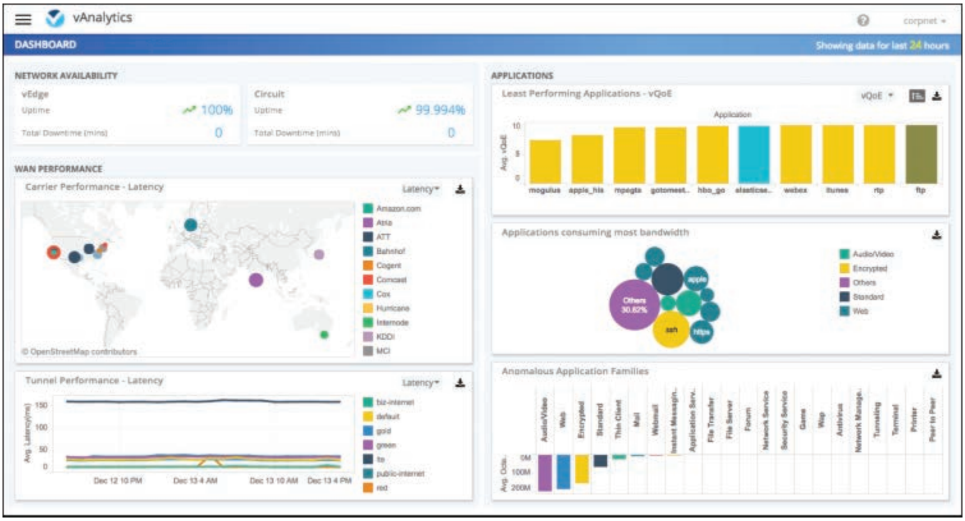
One of the keys to an SDN solution is the visibility into the network and the applications running over that network. The Cisco SD-WAN solution offers simple automation and analytics that give administrators valuable insights into network operations and performance.

The optional vAnalytics platform provides graphical representations of the performance of the entire Cisco SD-WAN overlay network over time and enables you to drill down to the characteristics of a single carrier, tunnel, or application at a particular time.

The vAnalytics dashboard provides an interactive overview of a network and serves as an entrance point for more details. The dashboard displays information for the past 24 hours. You have an option to drill down and select various time periods for which to display data.

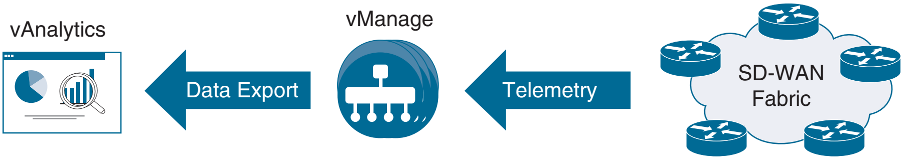
The vAnalytics platform displays application performance with the Quality of Experience (vQoE) value, as illustrated in Figure 17-6. This vQoE value ranges from 0 to 10, with 0 indicating the worst performance and 10 the best. The vAnalytics platform calculates the vQoE value based on latency, loss, and jitter, and it customizes the calculation for each application. Besides the vQoE values, the main dashboard displays network availability (uptime), carrier performance statistics, tunnel performance statistics, application bandwidth utilization, and anomalous application utilization.

Figure 17-6 Cisco SD-WAN vAnalytics


vAnalytics
corpnet
DASHBOARD
Showing data for last 24 hours
NETWORK AVAILABILITY
vEdge
Uptime
100%
Total Downtime (mins)
0
Circuit
Uptime
99.994%
Total Downtime (mins)
0
WAN PERFORMANCE
Carrier Performance - Latency
Latency
Amazon.com
Atla
ATT
Bahnhof
Cogent
Comcast
Cox
Hurricane
Internode
KDDI
MCI
© OpenStreetMap contributors
Tunnel Performance - Latency
Latency
Avg. Latency(ms)
150
100
50
0
Dec 12 10 PM
Dec 13 4 AM
Dec 13 10 AM
Dec 13 4 PM
biz-internet
default
gold
green
lte
public-internet
red
APPLICATIONS
Least Performing Applications - vQoE
vQoE
Application
Avg. vQoE
10
5
0
mogulus
apple_his
mpegts
gotomeet..
hbo_go
elasticse..
webex
itunes
rtp
ftp
Applications consuming most bandwidth
apple
Others
30.82%
ssh
https
Audio/Video
Encrypted
Others
Standard
Web
Anomalous Application Families
Avg. Octe
0M
100M
200M
Audio/Video
Web
Encrypted
Standard
Thin Client
Mail
Webmail
Instant Messagin..
Application Serv..
File Transfer
File Server
Forum
Network Service
Security Service
Game
Voip
Antivirus
Network Manage..
Tunneling
Terminal
Printer
Peer to Peer

As shown in Figure 17-7, data is collected by vManage and then exported securely to the vAnalytics platform. Only management data (statistics and flow information) is collected. No personally identifiable information (PII) is stored.

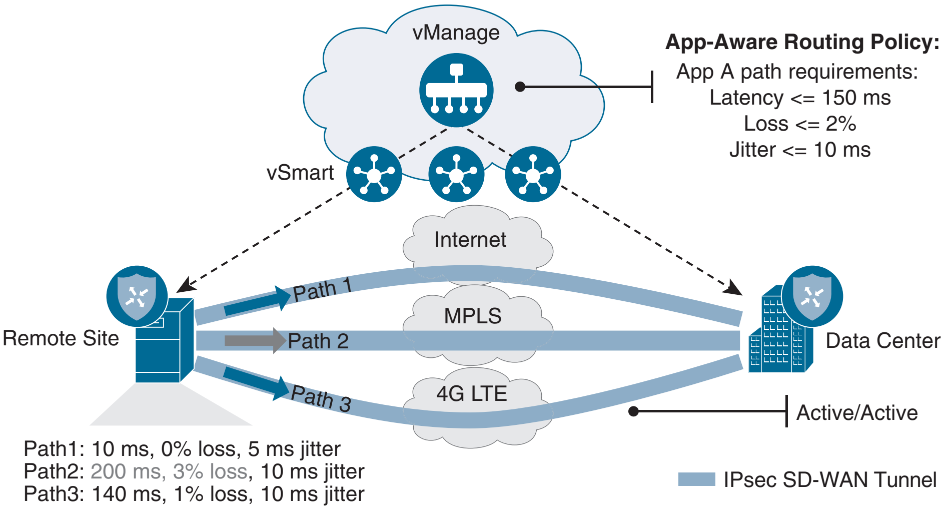
Figure 17-7 Cisco SD-WAN vAnalytics Information Flow


vAnalytics
Data Export
vManage
Telemetry
SD-WAN Fabric

Cisco SD-WAN Application Performance Optimization

A variety of network issues can impact application performance for end users, such as packet loss, congested WAN circuits, high-latency WAN links, and suboptimal WAN path selection. Optimizing the application experience is critical to achieving high user productivity. The Cisco SD-WAN solution can minimize loss, jitter, and delay and can overcome WAN latency and forwarding errors to optimize application performance. Figure 17-8 shows that for App A, Paths 1 and 3 are valid paths, but Path 2 does not meet the SLAs, so it is not used in path selection for transporting App A traffic. WAN Edge routers continuously perform path liveliness and quality measurements with Bidirectional Forwarding Detection (BFD).

Figure 17-8 Cisco SD-WAN Application-Aware Routing


vManage
App-Aware Routing Policy:
App A path requirements:
Latency <= 150 ms
Loss <= 2%
Jitter <= 10 ms
vSmart
Internet
Path 1
MPLS
Path 2
4G LTE
Path 3
Remote Site
Data Center
Active/Active
Path1: 10 ms, 0% loss, 5 ms jitter
Path2: 200 ms, 3% loss, 10 ms jitter
Path3: 140 ms, 1% loss, 10 ms jitter
IPsec SD-WAN Tunnel

The following Cisco SD-WAN capabilities help address application performance optimization:

- **Application-aware routing:** Application-aware routing makes it possible to create customized SLA policies for traffic and measure real-time performance based on readings from BFD probes. The application traffic is directed to WAN links that support the SLAs for the application. During periods of performance degradation, the traffic can be directed to other paths if SLAs are exceeded.

- **Quality of service (QoS):** QoS includes classification, scheduling, queueing, shaping, and policing of traffic on WAN router interfaces. QoS is designed to minimize the delay, jitter, and packet loss in critical application flows.

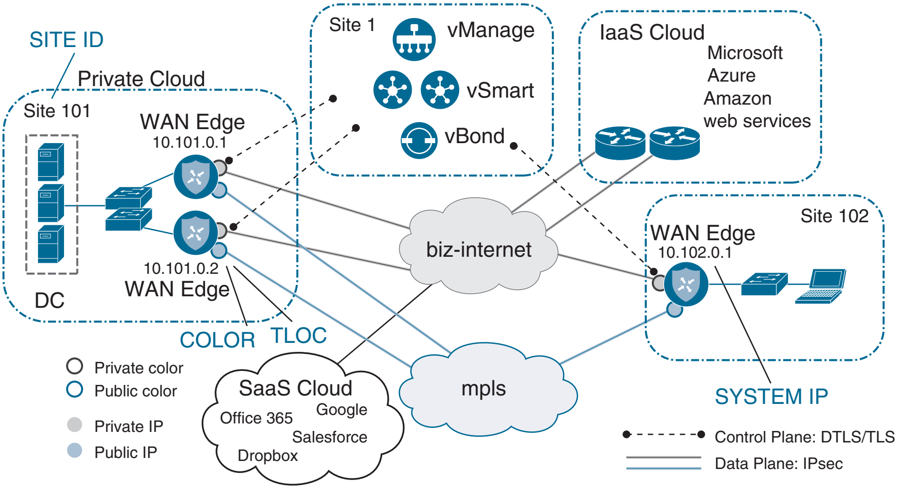
- **Software as a service (SaaS):** Traditionally, branches have accessed SaaS applications (such as Salesforce, Dropbox, and Office 365) through centralized data centers, and increased application latency and unpredictable user experience have resulted. As Cisco SD-WAN has evolved, additional network paths to access SaaS applications are possible, including Direct Internet Access (DIA) and access through regional gateways or colocation sites. However, network administrators may have limited or no visibility into the performance of the SaaS applications from remote sites, so choosing the network path for accessing the SaaS applications in order to optimize the end-user experience can be problematic. In addition, when changes to the network or impairment occur, there may not be an easy way to move affected applications to an alternate path. Cloud onRamp for SaaS allows you to easily configure access to SaaS applications either directly from the Internet or through gateway locations. It continuously probes, measures, and monitors the performance of each path to each SaaS application and chooses the best-performing path based on loss and delay. If impairment occurs, SaaS traffic is dynamically and intelligently moved to the updated optimal path.

- **Infrastructure as a service (IaaS):** IaaS delivers network, compute, and storage resources to end users on demand in a public cloud (such as AWS or Azure) or over the Internet. Traditionally, for a branch to reach IaaS resources, there was no direct access to public cloud data centers, as they typically required access through a corporate data center or colocation site. In addition, there was a dependency on MPLS to reach IaaS resources at private cloud data centers, with no consistent segmentation or QoS policies from the branch to the public cloud. Cisco Cloud onRamp for IaaS is a feature that automates connectivity to workloads in the public cloud from the data center or branch. It automatically deploys in the public cloud WAN Edge router instances that become part of the Cisco SD-WAN overlay and establish data plane connectivity to the routers located in the data center or branch. It extends full Cisco SD-WAN capabilities into the cloud and extends a common policy framework across the Cisco SD-WAN fabric and cloud. Cisco Cloud onRamp for IaaS eliminates the need for traffic from Cisco SD-WAN sites to traverse the data center, thus improving the performance of the applications hosted in the public cloud. It also provides high availability and path redundancy to applications hosted in the cloud by deploying a pair of virtual routers in a transit VPC/VNET configuration, which is also very cost-effective.

Cisco SD-WAN Solution Example

Figure 17-9 demonstrates several aspects of the Cisco SD-WAN solution. This sample topology depicts two WAN Edge sites (DC Site 101 and Branch Site 102), each directly connected to a private MPLS transport and a public Internet transport. The cloud-based SD-WAN controllers at Site 1 (the two vSmart controllers, the vBond orchestrator, and the vManage server) are reachable directly through the Internet transport. In addition, the topology includes cloud access to SaaS and IaaS applications.

Figure 17-9 Cisco SD-WAN Solution Example Topology


SITE ID
Private Cloud
Site 101
WAN Edge
10.101.0.1
10.101.0.2
WAN Edge
DC
COLOR
TLOC
Site 1
vManage
vSmart
vBond
IaaS Cloud
Microsoft Azure
Amazon web services
biz-internet
Site 102
WAN Edge
10.102.0.1
SYSTEM IP
mpls
SaaS Cloud
Office 365
Google
Salesforce
Dropbox
Private color
Public color
Private IP
Public IP
Control Plane: DTLS/TLS
Data Plane: IPsec

The WAN Edge routers form a permanent Datagram Transport Layer Security (DTLS) or Transport Layer Security (TLS) control connection to the vSmart controllers and connect to both of the vSmart controllers over each transport (mpls and biz-internet). The routers also form a permanent DTLS or TLS control connection to the vManage server—but over just one of the transports. The WAN Edge routers securely communicate to other WAN Edge routers using IPsec tunnels over each transport. The Bidirectional Forwarding Detection (BFD) protocol is enabled by default and runs over each of these tunnels, detecting loss, latency, jitter, and path failures. The example in Figure 17-9 shows the following Cisco SD-WAN elements:

Site ID

A site ID is a unique identifier of a site in the SD-WAN overlay network with a numeric value 1 through 4294967295 (2^32 – 1) that identifies the source location of an advertised prefix. This ID must be configured on every WAN Edge device, including the controllers, and must be the same for all WAN Edge devices that reside at the same site. A site could be a data center, a branch office, a campus, or something similar. By default, IPsec tunnels are not formed between WAN Edge routers within the same site that share the same site ID.

System IP

A system IP address is a persistent, system-level IPv4 address that uniquely identifies the device independently of any interface addresses. It acts much like a router ID, so it doesn't need to be advertised or known by the underlay. It is assigned to the system interface that resides in VPN 0 and is never advertised. A best practice is to assign the system IP address to a loopback interface and advertise it in any service VPN. It can then be used as a source IP address for SNMP and for logging, which makes it easier to correlate network events with vManage information.

Organization Name

An organization name is a name that is assigned to the SD-WAN overlay. It is case-sensitive and must match the organization name configured on all the SD-WAN devices in the overlay. It is used to define the Organization Unit (OU) field to match in the certificate authentication process when an SD-WAN device is brought into the overlay network.

Public and Private IP Addresses

There are two types of addresses used in Cisco SD-WAN:

- **Private IP address:** On WAN Edge routers, the private IP address is the IP address assigned to the interface of the SD-WAN device. This is the pre-NAT address, and despite the name, it can be a public address (publicly routable) or a private address (according to RFC 1918).
- **Public IP address:** The public IP address is the post-NAT address detected by the vBond orchestrator via the Internet transport. This address can be either a public address (publicly routable) or a private address (according to RFC 1918).

In the absence of NAT, the private and public IP addresses of an SD-WAN device are the same.

TLOC

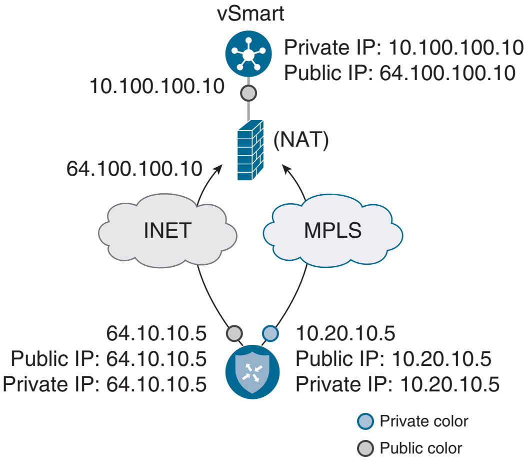
A TLOC, or transport location, is the attachment point where a WAN Edge router connects to the WAN transport network. A TLOC is uniquely identified and represented by a three-tuple, consisting of system IP address, link color, and encapsulation (Generic Routing Encapsulation [GRE] or IPsec). TLOC routes are advertised to vSmart controllers via OMP, along with a number of attributes, including the private and public IP addresses and port numbers associated with each TLOC, as well as color and encryption keys. These TLOC routes with their attributes are distributed to other WAN Edge routers. With the TLOC attributes and encryption key information known, the WAN Edge routers can attempt to form BFD sessions using IPsec with other WAN Edge routers. By default, WAN Edge routers attempt to connect to every TLOC over each WAN transport, including TLOCs that belong to other transports marked with different colors. This is helpful, for example, when you have different Internet transports at different locations that should communicate directly with each other.

Color

The color attribute applies to WAN Edge routers or vManage and vSmart controllers and helps identify an individual TLOC; different TLOCs are assigned different color labels. The SD-WAN topology in Figure 17-9 uses a public color called biz-internet for the Internet transport TLOC and a private color called mpls for the other transport TLOC. You cannot use the same color twice on a single WAN Edge router.

Figure 17-10 illustrates the concept of color and public/private IP addresses in Cisco SD-WAN. A vSmart controller interface is addressed with a private (RFC 1918) IP address, but a firewall translates that address into a publicly routable IP address that WAN Edge routers use to reach it. The figure also shows a WAN Edge router with an MPLS interface configured with an RFC 1918 IP address and an Internet interface configured with a publicly routable IP address. Since there is no NAT translating the private IP addresses of the WAN Edge router, the public and private IP addresses are the same in both cases. The transport color on the vSmart controller is set to a public color, and on the WAN Edge, the Internet side is set to a public color and the MPLS side is set to a private color. The WAN Edge router reaches the vSmart controller on either transport by using the remote public IP address (64.100.100.10) as the destination due to the public color on the vSmart interface.

Figure 17-10 Cisco SD-WAN Private and Public Colors



Overlay Management Protocol (OMP)

The OMP routing protocol, which has a structure similar to BGP, manages the SD-WAN overlay network. The protocol runs between vSmart controllers and between vSmart controllers and WAN Edge routers, where control plane information such as route prefixes, next-hop routes, crypto keys, and policy information is exchanged over a secure DTLS or TLS connection. The vSmart controller acts much like a BGP route reflector: It receives routes from WAN Edge routers, processes and applies any policy to them, and then advertises the routes to other WAN Edge routers in the overlay network.

Virtual Private Networks (VPNs)

In the SD-WAN overlay, virtual private networks (VPNs) provide segmentation, much like virtual routing and forwarding instances (VRFs). The VPNs are isolated from one another, and each VPN has its own forwarding table. An interface or a subinterface is explicitly configured under a single VPN and cannot be part of more than one VPN. Labels are used in OMP route attributes and in the packet encapsulation, which identifies the VPN a packet belongs to.

The VPN number is a 4-byte integer with a value from 0 to 65535, but several VPNs are reserved for internal use, so the maximum VPN that can or should be configured is 65527. There are two main VPNs present by default in the WAN Edge devices and controllers: VPN 0 and VPN 512. Note that VPN 0 and 512 are the only VPNs that can be configured on vManage and vSmart controllers. For the vBond orchestrator, although more VPNs can be configured, only VPN 0 and 512 are functional, and they are the only ones that should be used:

- VPN 0 is the transport VPN. It contains the interfaces that connect to the WAN transports. Secure DTLS/TLS connections to the controllers are initiated from this VPN. Static or default routes or a dynamic routing protocol needs to be configured inside this VPN in order to get appropriate next-hop information so the control plane can be established and IPsec tunnel traffic can reach remote sites.

- VPN 512 is the management VPN. It carries the out-of-band management traffic to and from the Cisco SD-WAN devices. This VPN is ignored by OMP and is not carried across the overlay network.

In addition to the default VPNs that are already defined, one or more service-side VPNs need to be created that contain interfaces that connect to the local-site network and carry user data traffic. It is recommended to select service VPNs in the range 1 to 511, but higher values can be chosen as long as they do not overlap with default and reserved VPNs. Service VPNs can be enabled for features such as OSPF or BGP, Virtual Router Redundancy Protocol (VRRP), QoS, traffic shaping, or policing. It is possible to direct user traffic over the IPsec tunnels to other sites by redistributing OMP routes received from the vSmart controllers at the site into the service-side VPN routing protocol. In turn, it is possible to advertise routes from the local site to other sites by advertising the service VPN routes into the OMP routing protocol, which is sent to the vSmart controllers and redistributed to the other WAN Edge routers in the network.

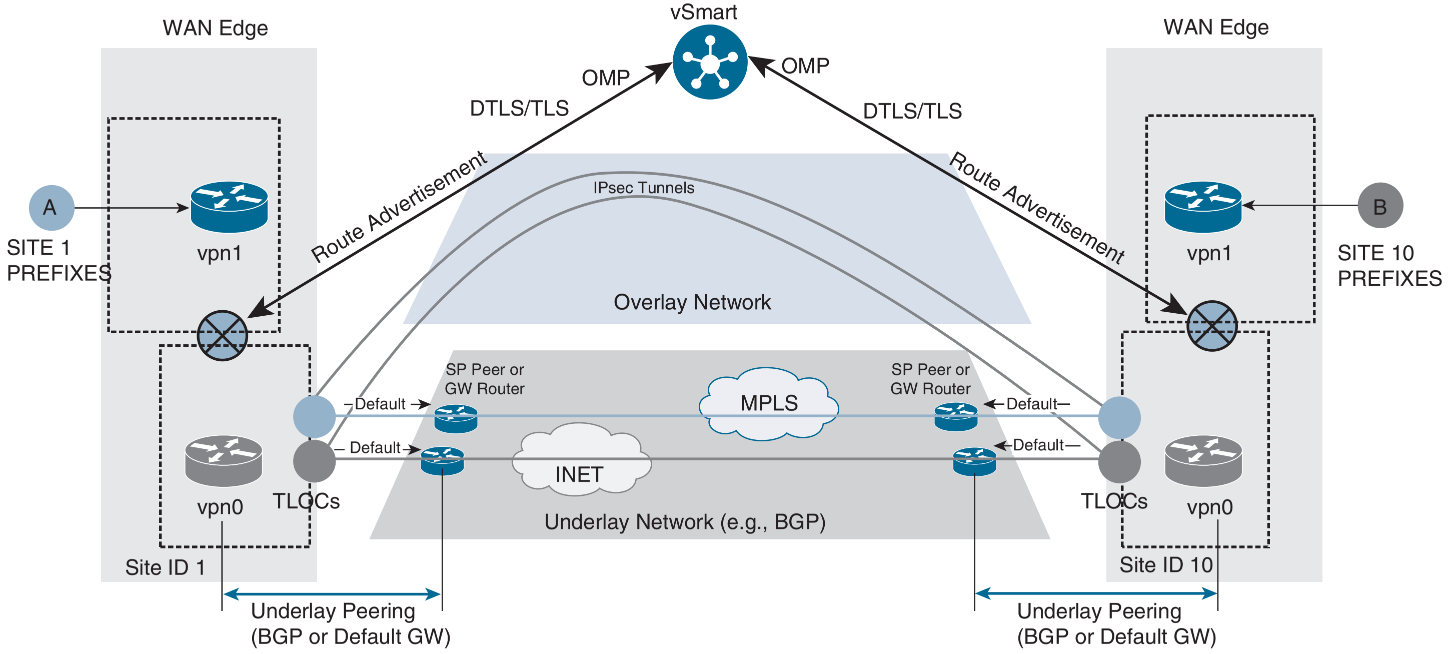
Cisco SD-WAN Routing

The Cisco SD-WAN network is divided into two distinct parts: the underlay and overlay networks. The underlay network is the physical network infrastructure that connects network devices such as routers and switches together and routes traffic between devices using traditional routing mechanisms. In the SD-WAN network, the underlay is typically made up of the connections from the WAN Edge router to the transport network and the transport network itself. The network ports that connect to the underlay network are part of VPN 0, the transport VPN. Getting connectivity to the service provider gateway in the transport network usually involves configuring a static default gateway (most common) or configuring a dynamic routing protocol, such as BGP or OSPF. These routing processes for the underlay network are confined to VPN 0, and their primary purpose is to provide reachability to TLOCs on other WAN Edge routers so that IPsec tunnels can be built to form the overlay network.

The IPsec tunnels that traverse from site to site by using the underlay network help to form the SD-WAN overlay fabric network. Overlay Management Protocol (OMP), a TCP-based protocol similar to BGP, provides the routing for the overlay network. The protocol runs between vSmart controllers and WAN Edge routers, and control plane information is exchanged over secure DTLS or TLS connections. The vSmart controller acts a lot like a route reflector: It receives routes from WAN Edge routers, processes and applies any policy to them, and then advertises the routes to other WAN Edge routers in the overlay network.

Figure 17-11 illustrates the relationship between OMP routing across the overlay network and BGP routing across the underlay. OMP runs between WAN Edge routers and vSmart controllers and also as a full mesh between vSmart controllers. When DTLS/TLS control connections are formed, OMP is automatically enabled.

Figure 17-11 **Cisco SD-WAN Routing with OMP**


vSmart
WAN Edge
WAN Edge
OMP
OMP
DTLS/TLS
DTLS/TLS
Route Advertisement
Route Advertisement
A
SITE 1
PREFIXES
B
SITE 10
PREFIXES
vpn1
vpn1
vpn0
vpn0
IPsec Tunnels
Overlay Network
SP Peer or
GW Router
SP Peer or
GW Router
Default
Default
Default
Default
MPLS
INET
TLOCs
TLOCs
Underlay Network (e.g., BGP)
Site ID 1
Site ID 10
Underlay Peering
(BGP or Default GW)
Underlay Peering
(BGP or Default GW)

OMP peering is established using the system IP addresses, and only one peering session is established between a WAN Edge device and a vSmart controller, even if multiple DTLS/TLS connections exist. OMP exchanges route prefixes, next-hop routes, crypto keys, and policy information.

OMP advertises three types of routes from WAN routers to vSmart controllers:

- **OMP routes:** OMP routes, or vRoutes, are prefixes that are learned from the local site, or service side, of a WAN Edge router. The prefixes are originated as static or connected routes or from within the OSPF, BGP, or EIGRP protocol and redistributed into OMP so they can be carried across the overlay. OMP routes advertise attributes such as transport location (TLOC) information, which is similar to a BGP next-hop IP address for the route, and other attributes, such as origin, origin metric, originator, preference, site ID, tag, and VPN. An OMP route is installed in the forwarding table only if the TLOC to which it points is active.
- **TLOC routes:** TLOC routes advertise TLOCs connected to the WAN transports, along with an additional set of attributes, such as TLOC private and public IP addresses, carrier, preference, site ID, tag, weight, and encryption key information.
- **Service routes:** Service routes represent services (for example, firewall, IPS, application optimization) that are connected to the WAN Edge local-site network and are available for other sites for use with service insertion. In addition, these routes include the originator system IP address, TLOC, and VPN ID; the VPN labels are sent in this update type to tell the vSmart controllers what VPNs are serviced at a remote site.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	23
CCNP Enterprise Design ENSLD 300-420 Official Cert Guide: Designing Cisco Enterprise Networks	11
Cisco SD-WAN Design Guide	https://www.cisco.com/c/en/us/td/docs/solutions/CVD/SDWAN/cisco-sdwan-design-guide.html
Cisco SD-WAN End-to-End Deployment Guide	https://www.cisco.com/c/dam/en/us/td/docs/solutions/CVD/SDWAN/SD-WAN-End-to-End-Deployment-Guide.pdf
Cisco Software-Defined Wide-Area Networks	Designing, Deploying, and Securing Your Next Generation WAN with Cisco SD-WAN

Day 16

Multicast

ENCOR 350-401 Exam Topics

Infrastructure

- IP Services
  - Describe multicast protocols, such as PIM and IGMP v2/v3

Key Topics

Today we review the benefits of IP multicast, explore typical applications that use multicast, and examine multicast addresses. We look at different versions of Internet Group Management Protocol (IGMP) and investigate basic Protocol Independent Multicast (PIM) features. We also explore multicast traffic flow from sender to receiver.

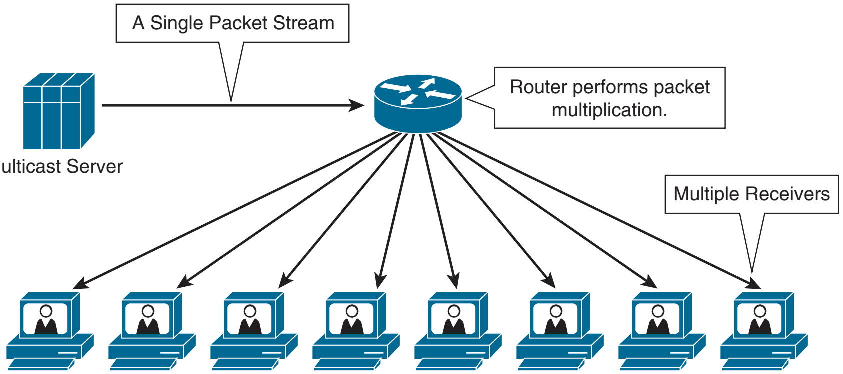
IP multicast has fundamentally changed the way that we consume content. This bandwidth conservation technology reduces traffic and server loads by simultaneously delivering a single stream of information to thousands of users. Applications that take advantage of multicast technologies include video conferencing, corporate communications, distance learning, distribution of software, stock quotes, and news.

Multicast Overview

There are three data communication methods in IPv4 networks: unicast, broadcast, and multicast. Unicast is usually referred to as a one-to-one communication method, broadcast is a one-to-all transmission method, and multicast is a one-to-many approach. Multicast is used to send the same data packets to multiple receivers. Sending to multiple receivers at once means the packets do not have to be duplicated for each receiver. Instead, they are sent in a single stream, and downstream routers perform packet multiplication over receiving links. Routers process fewer packets because each receives only a single copy of the packet. This packet is then multiplied and sent on outgoing interfaces where there are receivers, as illustrated in Figure 16-1.

Because downstream routers perform packet multiplication and delivery to receivers, the sender or source of multicast traffic does not have to know the unicast addresses of the receiver. Simulcast, which is simultaneous delivery for a group of receivers, may be used for several purposes, including audio and video streaming, news and similar data delivery, and software upgrade deployment.

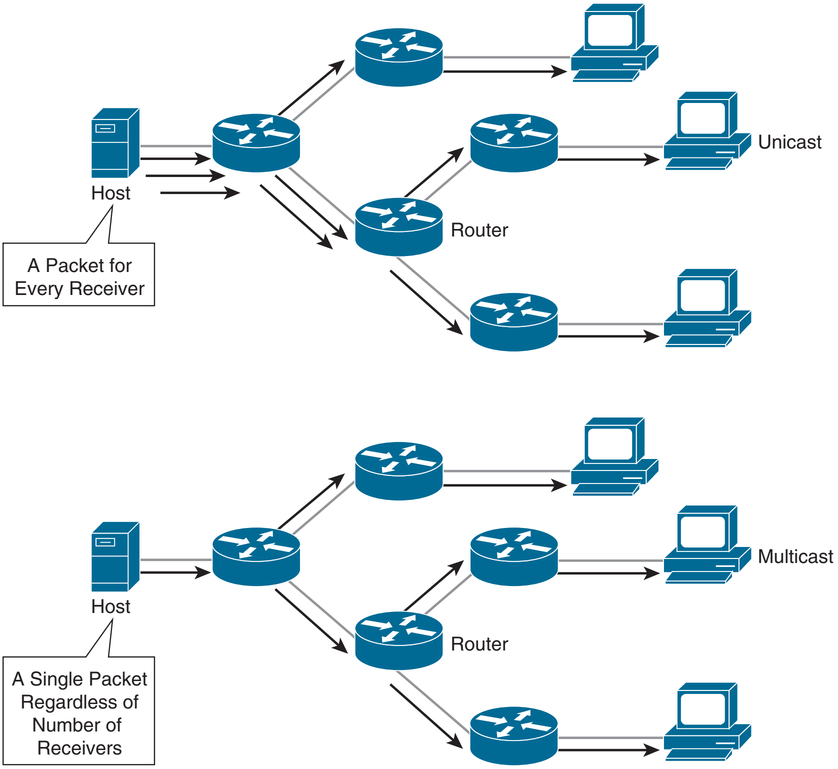
Figure 16-1 Multicast Communication


A Single Packet Stream
Router performs packet multiplication.
Multicast Server
Multiple Receivers

Unicast vs. Multicast

Unicast transmission involves sending multiple copies of data, one copy for each receiver. In other words, in unicast, the source sends a separate copy of a packet to each destination host that needs the information. Multicast transmission involves sending a single copy of data to multiple receivers. These processes are illustrated in Figure 16-2.

Figure 16-2 Unicast vs. Multicast Traffic Streams


Host
A Packet for Every Receiver
Router
Unicast
Host
A Single Packet Regardless of Number of Receivers
Router
Multicast

The upper part of the figure shows a host transmitting three copies of data and a network forwarding each packet to three separate receivers. The host may send to only one receiver at a time because it must create a different packet destination address for each receiver.

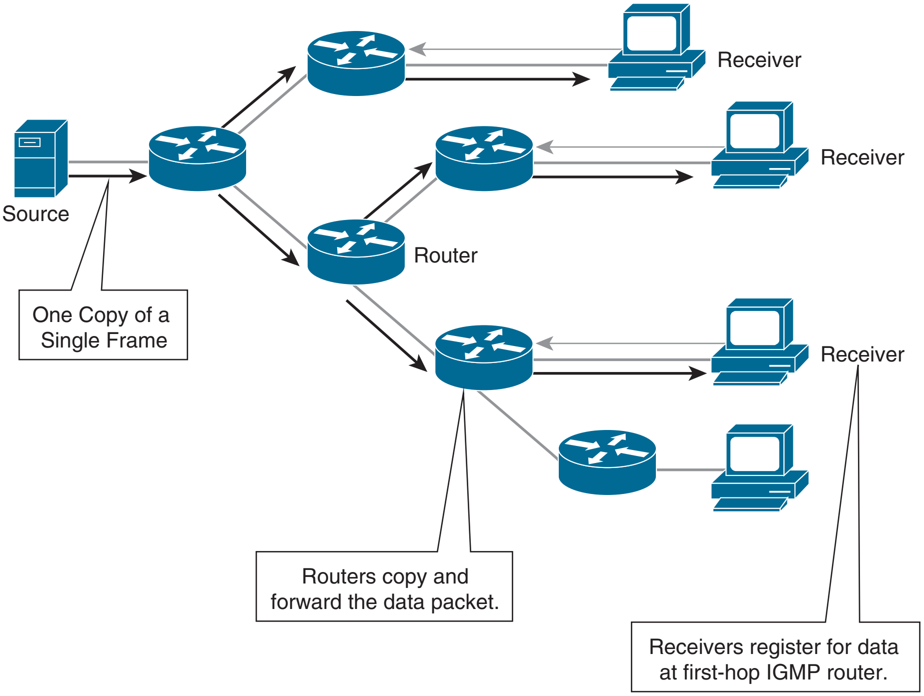
The lower part of the figure shows a host transmitting one copy of data and the network replicating the packet at the last possible hop for each receiver. Each packet exists only as a single copy on any given network. The host may send to multiple receivers simultaneously because it is sending only one packet.

Multicast Operations

In multicast, the source sends only one copy of a single data packet that is addressed to a group of receivers—a multicast group. Downstream multicast routers replicate and forward the data packet to all the branches where receivers exist. Receivers express their interest in multicast traffic by registering at their first-hop router using IGMP.

Figure 16-3 shows a multicast source host transmitting one copy of data and a network replicating the packet. Routers are responsible for replicating the packet and forwarding it to multiple recipients. Routers replicate the packet at any point where the network paths diverge, and they use reverse-path forwarding (RPF) techniques to ensure that the packet is forwarded to the appropriate downstream paths without routing loops. Each packet exists only as a single copy on any given network. The multicast source host may send to multiple receivers simultaneously because it is sending only one packet.

Figure 16-3 Multicast Forwarding

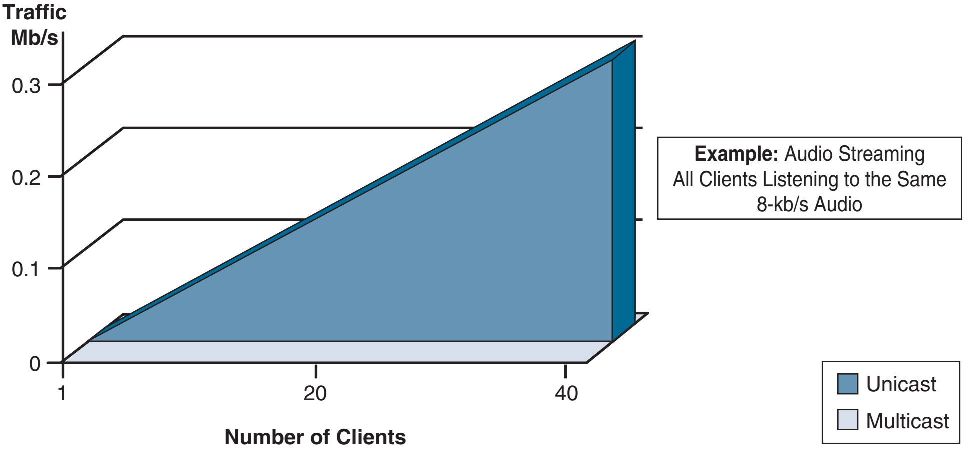

Receiver
Receiver
Source
Router
One Copy of a Single Frame
Receiver
Routers copy and forward the data packet.
Receivers register for data at first-hop IGMP router.

Multicast Benefits and Drawbacks

Multicast transmission provides many benefits over unicast transmission. The network experiences enhanced efficiency since the available network bandwidth is utilized more efficiently because multiple streams of data are replaced with a single transmission. Network devices have optimized performance due to fewer copies of data needing to be forwarded and processed. For the equivalent amount of multicast traffic, the sender needs much less processing power and bandwidth. Multicast packets do not impose high bandwidth utilization as unicast packets do, so there is a greater possibility that they will arrive almost simultaneously at the receivers. A whole range of new applications that were not possible on unicast (for example, IPTV) are available with multicast.

Distributed applications—that is, software running on multiple computers in the network at the same time and that can be stored with cloud computing or on servers—become available with multicast. Multipoint applications are not possible as demand and usage grows because unicast transmission does not scale. Traffic level and clients increase at a 1:1 rate with unicast transmission. Multicast does not have this limiting factor. Figure 16-4 shows that the bandwidth utilization for a multicast audio stream remains the same regardless of the number of clients.

Figure 16-4 Multicast and Unicast Bandwidth Utilization



Most multicast applications are UDP based. This foundation results in some undesirable consequences when compared to similar unicast TCP applications. UDP best-effort delivery results in occasional packet drops. These losses may affect many multicast applications that operate in real time (for example, video and audio). Also, requesting retransmission of the lost data at the application layer in these not-quite-real-time applications is not feasible. Heavy drops on voice applications result in jerky, missed speech patterns that can make the content unintelligible when the drop rate gets high enough. Sometimes, moderate to heavy drops in video appear as unusual artifacts in the picture. However, even low drop rates can severely affect some compression algorithms. This action causes the picture to freeze for several seconds while the decompression algorithm recovers.

IP Multicast Applications

There are two common multicast models:

- **One-to-many model:** In this model, one sender sends data to many receivers. Typical applications include video and audio broadcast.
- **Many-to-many model:** In this model, a host can simultaneously be a sender and a receiver. Typical applications include document sharing, group chat, and multiplayer games.

Other models (for example, many-to-one, where many receivers are sending data back to one sender, or few-to-many) are also used, especially in financial applications and networks.

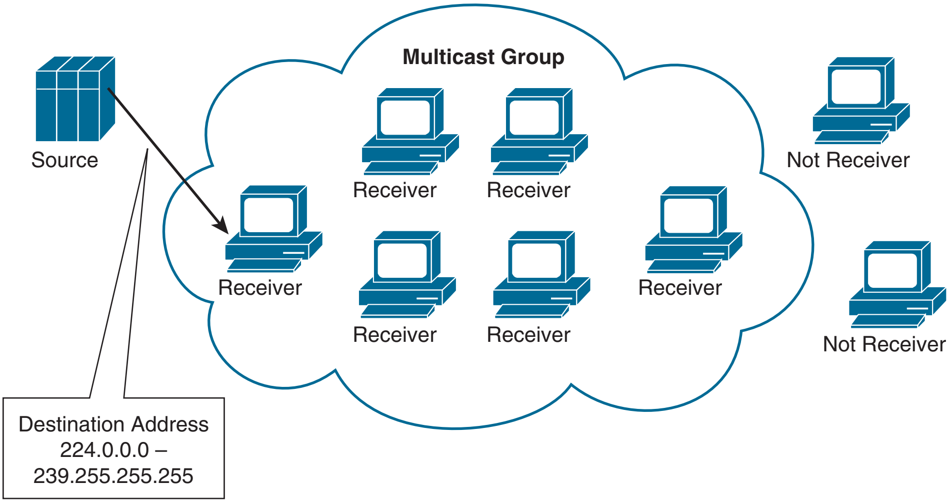
Many new multipoint applications are emerging as demand for them grows. Examples of these applications include the following:

- Real-time applications include live broadcasts, financial data delivery, whiteboard collaboration, and video conferencing.
- Non-real-time applications include file transfer, data and file replication, and VoD (video on demand).

IP Multicast Group Address

A multicast address is associated with a group of interested receivers. RFC 5771 designates addresses 224.0.0.0 through 239.255.255.255 as multicast addresses in IPv4. The sender sends a single datagram (from the sender's unicast address) to the multicast address, and the intermediary routers take care of making copies and sending them to all receivers that have registered their interest in data from that sender, as illustrated in Figure 16-5.

Figure 16-5 Multicast Group



Multicast IP addresses use the Class D address space. Class D addresses have the 4 high-order bits set to 1110. The multicast IPv4 address space is separated into several address groups, as shown in Table 16-1.

Table 16-1 IPv4 Multicast Address Space

Address Range	Size	Designation
224.0.0.0–224.0.0.255	/24	Local Network Control block
224.0.1.0–224.0.1.255	/24	Internetwork Control block
224.0.2.0–224.0.255.255	65024	Ad hoc block I
224.1.0.0–224.1.255.255	/16	Reserved
224.2.0.0–224.2.255.255	/16	SDP/SAP block
224.3.0.0–224.4.255.255	2 × /16	Ad hoc block II
224.5.0.0–224.255.255.255	251 × /16	Reserved
225.0.0.0–231.255.255.255	7 × /8	Reserved
232.0.0.0–232.255.255.255	/8	Source-Specific Multicast block
233.0.0.0–233.251.255.255	16515072	GLOP block
233.252.0.0–233.255.255.255	/14	Ad hoc block III
234.0.0.0–238.255.255.255	5 × /8	Reserved
239.0.0.0–239.255.255.255	/8	Administratively Scoped block

The following list briefly explains each block type and its proper usage:

- **Local Network Control block (224.0.0/24):** The local control block is used for specific protocol control traffic. Router interfaces listen to but do not forward local control multicasts. Assignments in this block are publicly controlled by IANA. Table 16-2 summarizes some of the well-known local network control multicast addresses.

Table 16-2 IPv4 Well-Known Local Network Control Multicast Addresses

Address	Description	References
224.0.0.1	All systems on this subnet	RFC 1112
224.0.0.2	All routers on this subnet	RFC 1112
224.0.0.5	OSPF (AllSPFRouters)	RFC 2328
224.0.0.6	OSPF (AllDRouters)	RFC 2328
224.0.0.9	RIPv2 routers	RFC 1723
224.0.0.10	EIGRP routers	RFC 7868
224.0.0.13	All PIM routers	RFC 7761
224.0.0.18	VRRP	RFC 5798
224.0.0.22	IGMPv3	RFC 3376
224.0.0.102	HSRPv2	Cisco

- **Internetwork Control block (224.0.1/24):** The Internetwork Control block is for protocol control traffic that router interfaces may forward through the autonomous system or through the Internet. Internetwork Control group assignments are also publicly controlled by IANA. Table 16-3 lists some of the well-known internetwork control multicast addresses.

Table 16-3 IPv4 Well-Known Internetwork Control Multicast Addresses

Address	Description	References
224.0.1.1	NTP	RFC 1119 and RFC 5905
224.0.1.39	cisco-rp-announce	Cisco
224.0.1.40	cisco-rp-discovery	Cisco

- **Ad hoc blocks (I: 224.0.2.0–224.0.255.255, II: 224.3.0.0–224.4.255.255, and III:233.252.0.0–233.255.255.255):** These blocks have traditionally been assigned to applications that do not fit in either the Local or Internetwork Control blocks. Router interfaces may forward ad hoc packets globally. Most applications using Ad hoc blocks require few group addresses (for example, less than a /24 space). IANA controls any public Ad hoc block assignments, and future assignments will come from Ad hoc block III if they are not more suited to Local Control or Internetwork Control blocks. Public use of unassigned ad hoc space is also permitted.
- **SDP/SAP block (224.2.0.0/16):** The Session Description Protocol/Session Announcement Protocol (SDP/SAP) block is assigned to applications that receive addresses through SAP, as described in RFC 2974.
- **Source-Specific Multicast block (232.0.0.0/8):** SSM addressing is defined by RFC 4607. SSM is a group model of IP multicast in which multicast traffic is forwarded to receivers from only those multicast sources for which the receivers have explicitly expressed interest. SSM is mostly used in one-to-many applications. No official assignment from IANA is required to use the SSM block because the application is local to the host; however, according to IANA policy, the block is explicitly reserved for SSM applications and must not be used for any other purpose.
- **GLOP block (233.0.0.0/8):** These addresses are statically assigned with a global scope. Each GLOP static assignment corresponds to a domain with a public 16-bit autonomous system number (ASN), which is issued by IANA. The ASN is inserted in dotted-decimal into the middle two octets of the multicast group address (X.Y). An example of GLOP assignment for an ASN of X.Y would be 233.X.Y.0/24. Domains using an assigned 32-bit ASN should apply for group assignments from the Ad hoc block III. Another alternative is to use IPv6 multicast group addressing. Because the ASN is public, IANA does not need to assign the GLOP groups. The GLOP block is intended for use by public content, network, and Internet service providers.
- **Administratively Scoped block (239.0.0.0/8):** Administratively scoped addresses are intended for local use within a private domain, as described by RFC 2365. These group addresses serve a similar function to RFC 1918 private IP addresses (for example, 10.0.0.0/8 or 172.16–31.0.0/16 blocks). Network architects can create an address schema using this block that best suits the needs of the private domain and can further split scoping into specific geographies, applications, or networks.

IP Multicast Service Model

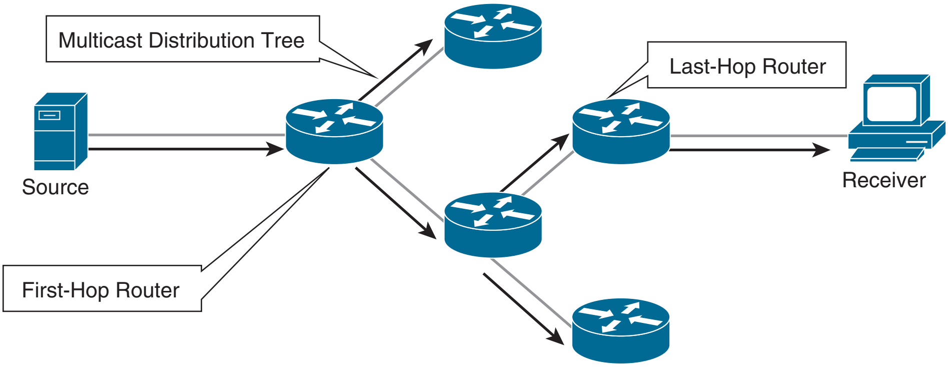
IP multicast service models consist of three main components: Senders send to a multicast address, receivers express an interest in a multicast address, and routers deliver traffic from the senders to the receivers. Each multicast group is identified by a Class D IP address. Members join and leave the group and indicate these activities to the routers. Routers listen to all multicast addresses and use multicast routing protocols to manage groups.

RFC 1112 specifies the host extensions for IP to support multicast:

- IP multicast allows hosts to join a group that receives multicast packets.
- It allows users to dynamically register (join or leave multicast groups) based on the applications they use.
- It uses IP datagrams to transmit data.

Receivers may dynamically join or leave an IPv4 multicast group at any time by using IGMP (Internet Group Management Protocol) messages, and they may dynamically join or leave an IPv6 multicast group at any time by using MLD (Multicast Listener Discovery) messages. Messages are sent to the multicast last-hop routers, which manage group membership, as illustrated in Figure 16-6.

Figure 16-6 Multicast Traffic Distribution



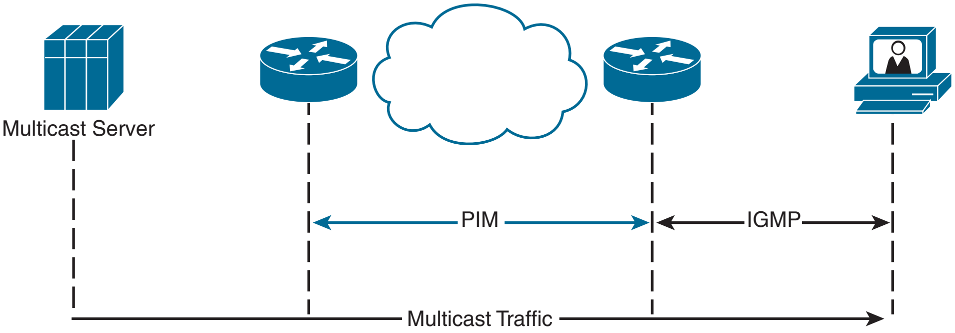
Routers use multicast routing protocols—for example, PIM (Protocol Independent Multicast)—to efficiently forward multicast data to multiple receivers. The routers listen to all multicast addresses and create multicast distribution trees, which are used for multicast packet forwarding.

Routers identify multicast traffic and forward the packets from senders toward the receivers. When the source becomes active, it starts sending the data without any indication. First-hop routers, to which the sources are directly connected, start forwarding the data toward the network. Receivers that are interested in receiving IPv4 multicast data register to the last-hop routers using IGMP membership messages. Last-hop routers are routers that have directly connected receivers. Last-hop routers forward the group membership information of their receivers to the network so that the other routers are informed about which multicast flows are needed. Figure 16-6 shows a multicast source that is connected to a first-hop router, which forwards multicast packets into the network. Packets traverse the shortest path tree on their way to the receivers toward the last-hop router.

Internet Group Management Protocol

The primary purpose of Internet Group Management Protocol (IGMP) is to permit hosts to communicate their desire to receive multicast traffic to the IP multicast router on the local network. This action, in turn, permits the IP multicast router to join the specified multicast group and to begin forwarding the multicast traffic onto the network segment. Figure 16-7 shows where IGMP is used in a network.

Figure 16-7 IGMP in Multicast Architecture



The initial specification for IGMPv1 was documented in RFC 1112. Since that time, many problems and limitations with IGMPv1 have been discovered, and in response, newer IGMP specifications have been developed. IGMPv2 is defined in RFC 2236. The latest version of IGMP, IGMPv3, is defined in RFC 3376.

IGMPv1 Overview

RFC 1112 specifies IGMP as a protocol used by IP hosts to report multicast group membership to their first-hop multicast routers. It uses a query/response model. Multicast routers periodically (usually every 60 to 120 seconds) send membership queries to the all-hosts multicast address (224.0.0.1) to solicit information on which multicast groups are active on the local network.

Hosts, wanting to receive specific multicast group traffic, send membership reports. Membership reports are sent (with a TTL of 1) to the multicast address of the group from which the host wants to receive traffic. Hosts either send reports asynchronously (when they want to first join a group—unsolicited reports) or in response to membership queries. In the latter case, the response is used to maintain the group in an active state so that traffic for the group remains forwarded to the network segment.

After a multicast router sends a membership query, there may be many hosts interested in receiving traffic from specified multicast groups. To suppress a membership report storm from all group members, a report suppression mechanism is used among group members. Report suppression saves CPU time and bandwidth on all systems.

Because membership query and report packets have only local significance, the TTL for these packets is always set to 1. TTL also must be set to 1 because forwarding of membership reports from a local subnet may cause confusion on other subnets.

If multicast traffic is forwarded on a local segment, there must be at least one active member of that multicast group on a local segment.

IGMPv2 Overview

Some limitations were discovered in IGMPv1. Most of the changes between IGMPv1 and IGMPv2 were made primarily to address the issues of leave and join latencies and also to address ambiguities in the original protocol specification. The following changes were made in revising IGMPv1 to IGMPv2:

- **Group-specific queries:** A group-specific query that was added in IGMPv2 allows the router to query its members only in a single group instead of querying all groups. This action is an optimized way to quickly find out if any members are left in a group without asking all groups for reports. The difference between the group-specific query and the membership query is that a membership query is multicast to the all-hosts address (224.0.0.1), whereas a group-specific query for group G is multicast to the group G multicast address.
- **Leave-group message:** A leave-group message allows hosts to tell the router that they are leaving the group. This information reduces the leave latency for the group on the segment when the member who is leaving is the last member of the group.
- **Querier election mechanism:** Unlike IGMPv1, IGMPv2 has a querier election mechanism. The lowest unicast IP address of the IGMPv2-capable routers is elected as the querier. By default, all IGMP routers are initialized as queriers, but they must immediately relinquish that role if a lower-IP-address query is heard on the same segment.
- **Query-interval response time:** The query-interval response time has been added to control the burstiness of reports. This time is set in queries to convey to the members how much time they must wait before they respond to a query with a report.

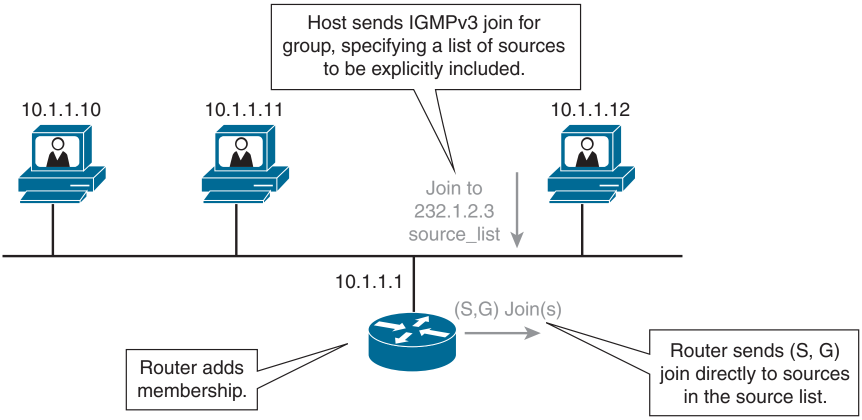
IGMPv2 is backward compatible with IGMPv1.

IGMPv3 Overview

IGMPv3, the next step in the evolution of IGMP, adds support for source filtering, which enables a multicast receiver host to signal to a router the groups from which it wants to receive multicast traffic and from which sources this traffic is expected. This membership information enables Cisco IOS Software to forward traffic from only those sources from which receivers requested the traffic. Although there are vast improvements with IGMPv3, backward compatibility still exists for the three versions.

Figure 16-8 shows IGMPv3 operation. The host 10.1.1.12 sends a join message with an explicit request to join group 232.1.2.3 but from a specific source (or sources), as listed in the *source_list* field in the IGMPv3 packet. The (S, G) message sent by the router indicates the required IP address of the multicast source, as well as the group multicast address. This type of message is forwarded using Protocol Independent Multicast (PIM).

Figure 16-8 IGMPv3 Join Message


Host sends IGMPv3 join for group, specifying a list of sources to be explicitly included.
10.1.1.10
10.1.1.11
10.1.1.12
Join to 232.1.2.3 source_list
10.1.1.1
(S,G) Join(s)
Router adds membership.
Router sends (S, G) join directly to sources in the source list.

Multicast Distribution Trees

Multicast-capable routers create distribution trees that control the path that IP multicast traffic takes through the network to deliver traffic to all receivers. The two basic types of multicast distribution trees are source trees and shared trees.

Source Trees

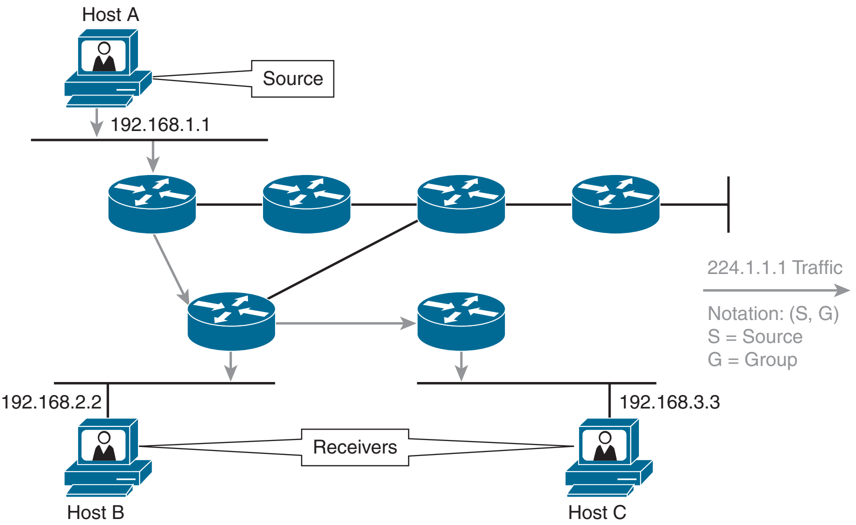
The simplest form of a multicast distribution tree is a *source tree*, with its root at the source and branches forming a spanning tree through the network to the receivers. Because this tree uses the shortest path through the network, it is also referred to as a shortest path tree (SPT).

Figure 16-9 shows an example of an SPT for group 224.1.1.1 rooted at the source, Host A, and connecting two receivers, Host B and Host C.

The special notation (S, G), pronounced "S comma G," enumerates an SPT where S is the IP address of the source, and G is the multicast group address. Using this notation, the SPT for the example shown in figure would be (192.168.1.1, 224.1.1.1).

The (S, G) notation implies that a separate SPT exists for each individual source sending to each group—and this is correct. For example, if Host B is also sending traffic to group 224.1.1.1 and Hosts A and C are receivers, a separate (S, G) SPT would exist, with the notation (192.168.2.2, 224.1.1.1).

Figure 16-9 Multicast Source Tree Example

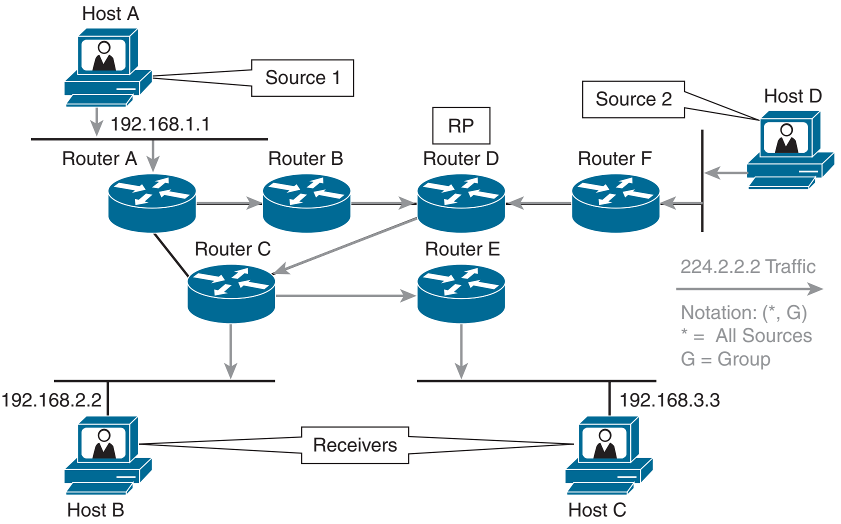

Host A
Source
192.168.1.1
224.1.1.1 Traffic
Notation: (S, G)
S = Source
G = Group
192.168.2.2
192.168.3.3
Receivers
Host B
Host C

Shared Trees

Unlike source trees that have their root at the source, shared trees use a single common root placed at some chosen point in the network. This shared root is called a rendezvous point (RP).

Figure 16-10 shows a shared tree for the group 224.2.2.2 with the root of the shared tree located at Router D. This shared tree is unidirectional. Source traffic is sent toward the RP on a source tree. The traffic is then forwarded down the shared tree from the RP to reach all receivers (unless the receiver is located between the source and the RP, in which case it is serviced directly).

Figure 16-10 Multicast Shared Tree Example


Host A
Source 1
192.168.1.1
RP
Source 2
Host D
Router A
Router B
Router D
Router F
Router C
Router E
224.2.2.2 Traffic
Notation: (*, G)
* = All Sources
G = Group
192.168.2.2
192.168.3.3
Receivers
Host B
Host C

In this example, multicast traffic from the sources, Hosts A and D, travels to the root (Router D) and then down the shared tree to the two receivers, Hosts B and C. Because all sources in the multicast group use a common shared tree, a wildcard notation, written as (*, G) and pronounced "star comma G," represents the tree. In this case, * means all sources, and G represents the multicast group. Therefore, the shared tree shown in the figure would be written as (*, 224.2.2.2).

Source Trees vs. Shared Trees

Both source trees and shared trees are loop free. Messages are replicated only where the tree branches.

Members of multicast groups can join or leave at any time; therefore, the distribution trees must be dynamically updated. When all the active receivers on a specific branch stop requesting the traffic for a specific multicast group, the routers prune that branch from the distribution tree and stop forwarding traffic down that branch. If one receiver on that branch becomes active and requests the multicast traffic, the router dynamically modifies the distribution tree and starts forwarding traffic again.

Source trees have the advantage of creating an optimal path between the source and the receivers. This advantage guarantees the minimum amount of network latency for forwarding multicast traffic. However, this optimization comes at a cost: The routers must maintain path information for each source. In a network that has thousands of sources and thousands of groups, this overhead can quickly become a resource issue on the routers. Memory consumption from the size of the multicast routing table is a factor that network designers must take into consideration.

Shared trees have the advantage of requiring the minimum amount of state in each router. This advantage reduces the overall memory requirements for a network that only allows shared trees. The disadvantage of shared trees is that under certain circumstances, the paths between the source and receivers might not be the optimal paths, and this might introduce some latency in packet delivery. For example, in Figure 16-10, the shortest path between Host A (source 1) and Host B (a receiver) would be Router A and Router C. Because Router D is the root for a shared tree, the traffic must traverse Routers A, B, D, and then C. Network designers must carefully consider the placement of the rendezvous point (RP) when implementing a shared tree environment.

IP Multicast Routing

In unicast routing, traffic is routed through the network along a single path from the source to the destination host. A unicast router does not consider the source address; it considers only the destination address and how to forward the traffic toward that destination. The router scans through its routing table for the destination address and then forwards a single copy of the unicast packet out the correct interface in the direction of the destination.

In multicast forwarding, the source is sending traffic to an arbitrary group of hosts that are represented by a multicast group address. The multicast router must determine which direction is the upstream direction (toward the source) and which is the downstream direction (or directions) toward the receivers. If there are multiple downstream paths, the router replicates the packet and forwards it down the appropriate downstream paths (using the best unicast route metric)—which is not necessarily all paths. The process of forwarding multicast traffic away from the source rather than to the receiver is called reverse-path forwarding (RPF). The basic idea of RPF is that when a multicast packet is received on the router interface, the router uses the source address to verify that the packet is not in a loop. The router checks the source IP address of the packet against the routing table, and if the interface that the routing table indicates is the same interface on which the packet was received, the packet passes the RPF check.

Protocol Independent Multicast

Protocol Independent Multicast (PIM) is protocol-independent IP routing and can leverage whichever unicast routing protocols are used to populate the unicast routing table, including Enhanced Interior Gateway Routing Protocol (EIGRP), Open Shortest Path First (OSPF), Border Gateway Protocol (BGP), and static routes. PIM uses this unicast routing information to perform the multicast forwarding function. Although PIM is called a multicast routing protocol, it actually uses the unicast routing table to perform the RPF check function instead of building up a completely independent multicast routing table. Unlike other routing protocols, PIM does not send and receive routing updates between routers.

There are two types of PIM multicast routing models: PIM Dense Mode (PIM-DM) and PIM Sparse Mode (PIM-SM). PIM-SM is the more commonly used model, and PIM-DM is not likely to be used.

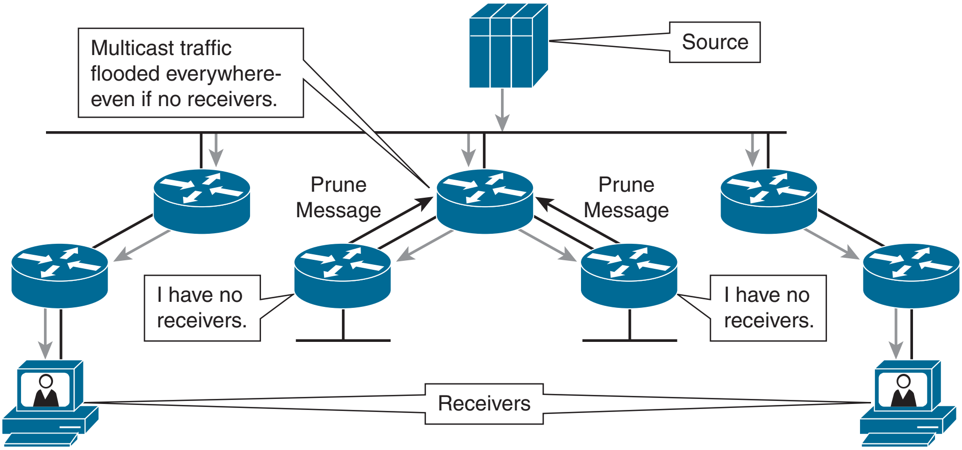
In Figure 16-7 earlier in this chapter, you can see that PIM operates between routers that are forwarding multicast traffic from the source to the receivers.

PIM-DM Overview

PIM-DM uses a push model to flood multicast traffic to every corner of the network. This push model is a brute-force method for delivering data to the receivers. This method would be efficient in certain deployments in which there are active receivers on every subnet in the network.

PIM-DM initially floods multicast traffic throughout the network. Routers that have no downstream neighbors prune back the unwanted traffic. This process, which repeats every 3 minutes, is illustrated in Figure 16-11.

Figure 16-11 PIM-DM Example



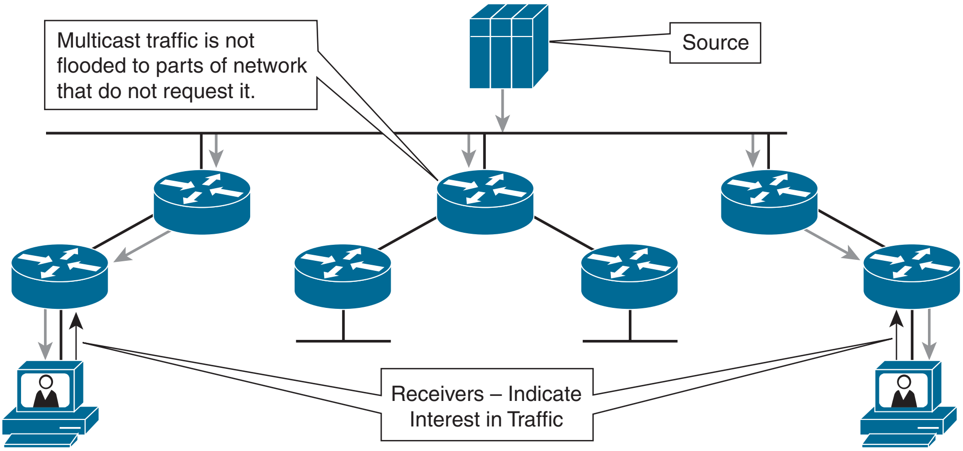
Routers accumulate state information by receiving data streams through the flood-and-prune mechanism. These data streams contain the source and group information so that downstream routers can build up their multicast forwarding table. PIM-DM supports only source trees—that is, (S, G) entries—and cannot be used to build a shared distribution tree.

PIM-SM Overview

PIM-SM, which is described in RFC 7761, operates independently of underlying unicast protocols. PIM-SM uses shared distribution trees rooted at the RP, but it may also switch to the source-rooted distribution tree.

PIM-SM is based on an explicit pull model, which means traffic is forwarded only to those parts of the network that need it, as illustrated in Figure 16-12.

Figure 16-12 PIM-SM Example


Multicast traffic is not flooded to parts of network that do not request it.
Source
Receivers – Indicate Interest in Traffic

PIM-SM uses an RP to coordinate the forwarding of multicast traffic from a source to its receivers. Senders register with the RP and send a single copy of multicast data through it to the registered receivers. Group members are joined to the shared tree by their local designated router. A shared tree that is built this way is always rooted at the RP.

PIM-SM is appropriate for wide-scale deployment of both densely and sparsely populated groups in the enterprise network. It is the optimal choice for all production networks, regardless of size and membership density.

There are many optimizations and enhancements to PIM, including the following:

- **Bidirectional PIM mode (BIDIR-PIM mode):** This mode is designed for many-to-many applications.
- **SSM:** This variant of PIM-SM only builds source-specific SPTs and does not need an active RP for source-specific groups.

Last-hop routers send PIM join messages to a designated RP. The RP is the root of a shared distribution tree down which all multicast traffic flows.

To get multicast traffic to the RP for distribution down the shared tree, first-hop routers with directly connected senders send PIM register messages to the RP. Register messages cause the RP to send an (S, G) join toward the first-hop router. This activity enables multicast traffic to flow natively to the RP via an SPT and, hence, down the shared tree.

Routers may be configured with an SPT threshold, and when this threshold is exceeded, it prompts the last-hop router to join the SPT. This action causes the multicast traffic from the first-hop router to flow down the SPT directly to the last-hop router.

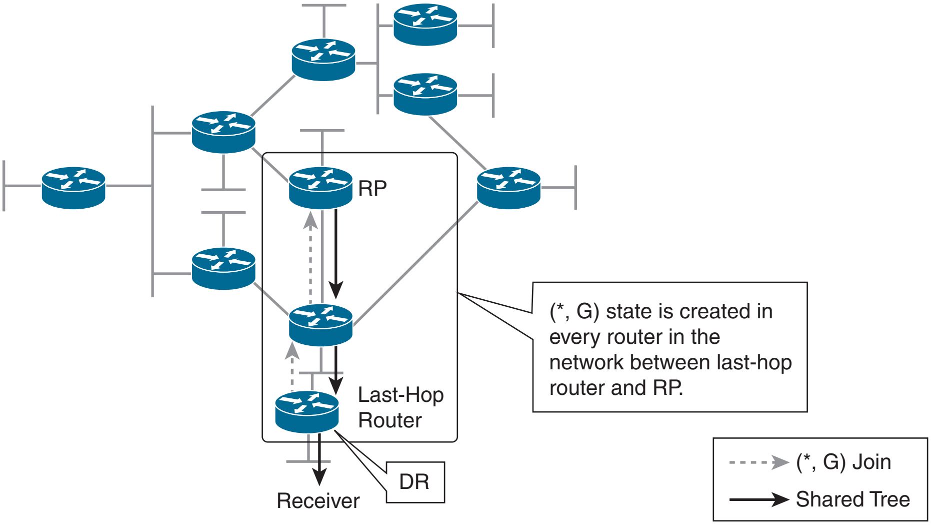
The RPF check is done differently depending on tree type. If traffic is flowing down the shared tree, the RPF check mechanism uses the IP address of the RP to perform the RPF check. If traffic is flowing down the SPT, the RPF check mechanism uses the IP address of the source to perform the RPF check.

Although it is common for a single RP to serve all groups, it is possible to configure different RPs for different groups or group ranges. This approach is accomplished via access lists. Access lists permit you to place the RPs in different locations in the network for different group ranges. The advantage to this approach is that it may improve or optimize the traffic flow for the different groups. However, only one RP for a group may be active at a time.

PIM-SM Shared Tree Join

In Figure 16-13, an active receiver has joined multicast group G by multicasting an IGMP membership report. A designated router on the LAN segment receives IGMP membership reports.

Figure 16-13 PIM-SM Shared Tree Join Example



(*, G) state is created in every router in the network between last-hop router and RP.

The designated router knows the IP address of the RP router for group G and sends a (*, G) join for this group toward the RP.

This (*, G) join travels hop by hop toward the RP, building a branch of the shared tree that extends from the RP to the last-hop router directly connected to the receiver.

At this point, group G traffic may flow down the shared tree to the receiver.

PIM-SM Sender Registration

When an active source for group G starts sending multicast packets, its first-hop designated router registers the source with the RP. To register a source, the first-hop router encapsulates the multicast packets in a PIM register message and sends the message to the RP using unicast.

After the shortest path tree is built from the first-hop router to the RP, the multicast traffic starts to flow from the source (S) to the RP without being encapsulated in register messages.

When the RP begins receiving multicast data down the shortest path tree from the source, it sends a PIM register-stop message to the first-hop router. The PIM register-stop message informs the first-hop router that it may stop sending the unicast register messages. At this point, the multicast traffic from the source is flowing down the shortest path tree to the RP and, from there, down the shared tree to the receivers.

Rendezvous Point

A rendezvous point (RP) is a router in a multicast network domain that acts as a shared root for a multicast shared tree. This section examines the different methods for deploying RPs. Any number of routers can be configured to work as RPs, and they can be configured to cover different group ranges. For correct operation, every multicast router within a Protocol Independent Multicast (PIM) domain must be able to map a specific multicast group address to the same RP.

Static RP

It is possible to statically configure an RP for a multicast group range. The address of the RP must be configured on every router in the domain.

Configuring static RPs is relatively easy and can be done with one or two lines of configuration on each router. If the network does not have many different RPs defined or if they do not change very often, then this could be the simplest method of defining RPs. This can also be an attractive option if the network is small. However, this can be a laborious task in a large and complex network. Every router must have the same RP address, which means changing the RP address requires reconfiguration of every router. If several RPs are active for different groups, information regarding which RP is handling each group must be known by all routers. To ensure that this information is complete, several configuration commands may be required. If the manually configured RP fails, there is no failover procedure for another router to take over the function performed by the failed RP. This method does not provide any kind of load balancing. Static RP can be combined with Anycast RP to provide RP load sharing and redundancy. PIM-SM, as defined in RFC 2362, allows for only a single active RP per group, and therefore the decision of optimal RP placement can become problematic for a multi-regional network deploying PIM-SM. Anycast RP relaxes an important constraint in PIM-SM—namely, that there can be only one group-to-RP mapping active at any time.

Static RP can coexist with dynamic RP mechanisms (such as Auto-RP). A dynamically learned RP takes precedence over manually configured RPs. If a router receives Auto-RP information for a multicast group that has manually configured RP information, the Auto-RP information is used.

PIM Bootstrap Router

The bootstrap router (BSR) is a mechanism for a router to learn RP information. It ensures that all routers in a PIM domain have the same RP cache as the BSR. It is possible to configure a BSR to help select an RP set from BSR candidate RPs. The function of the BSR is to broadcast the RP set to all routers in the domain.

The elected BSR receives candidate RP messages from all candidate RPs in the domain. The bootstrap message sent by the BSR includes information about all the candidate RPs. The routers use a common algorithm to select the same RP address for a given multicast group.

The BSR mechanism is a nonproprietary method of defining RPs that can be used with third-party routers (which support the BSR mechanism). It is not necessary to separately configure each router (except in the case of candidate BSRs and candidate RPs). The mechanism is largely self-configuring, which makes it easier to modify RP information. Information regarding several RPs for different groups is automatically communicated to all routers, which reduces administrative overhead. The mechanism is robust to router failover and permits backup RPs to be configured. If there is an RP failure, the secondary RP for the group can take over as the RP for the group.

Auto-RP and BSR protocols must not be configured together in the same network.

Auto-RP

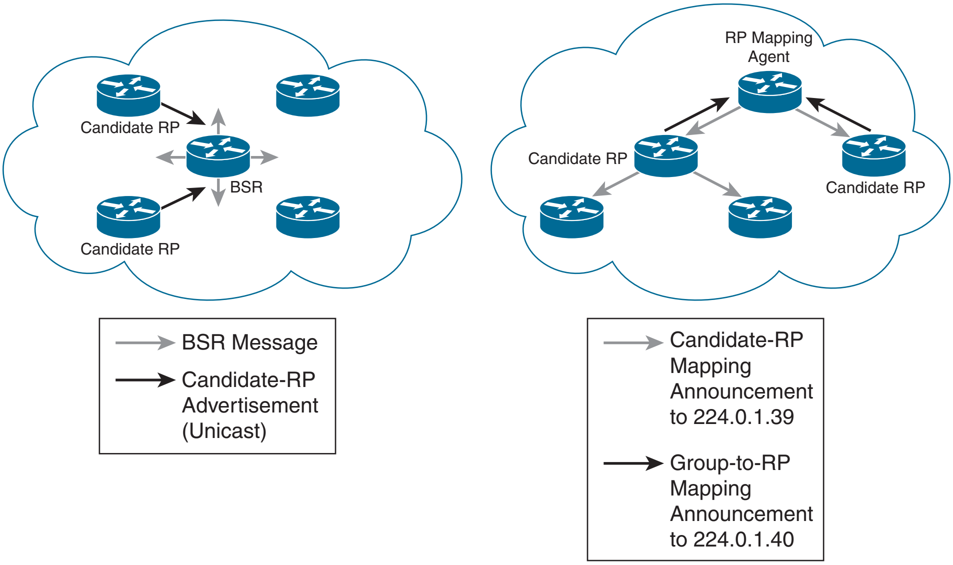
Auto-RP is a mechanism for automating distribution of RP information in a multicast network. The Auto-RP mechanism operates using two basic components:

- **Candidate RPs:** These RPs advertise their willingness to be an RP via RP-announcement messages, which are periodically sent to the reserved well-known group 224.0.1.39 (CISCO-RP-ANNOUNCE).
- **RP-mapping agents:** These agents join group 224.0.1.39 and map the RPs to the associated groups. The RP-mapping agents advertise the authoritative RP mappings to another well-known group address, 224.0.1.40 (CISCO-RP-DISCOVERY). All PIM routers join 224.0.1.40 and store the RP mappings in their private cache.

All routers automatically learn the RP information, which makes it easier to administer and update RP information. It is not necessary to separately configure each router (except for candidate RPs and mapping agents). Auto-RP permits backup RPs to be configured, which provides an RP failover mechanism. Auto-RP is a Cisco-proprietary mechanism.

BSR and Auto-RP protocols must not be configured together in the same network. Figure 16-14 illustrates the BSR and Auto-RP distribution mechanisms. The cloud on the left represents the BSR process, and the cloud on the right represents the Auto-RP process.

Figure 16-14 PIM RP Distribution Mechanisms


Candidate RP
BSR
Candidate RP
BSR Message
Candidate-RP Advertisement (Unicast)
RP Mapping Agent
Candidate RP
Candidate RP
Candidate-RP Mapping Announcement to 224.0.1.39
Group-to-RP Mapping Announcement to 224.0.1.40

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	13
IP Multicast, Volume 1: Cisco IP Multicast Networking	1, 2, 3, 4
CCNP Enterprise Design ENSLD 300-420 Official Cert Guide: Designing Cisco Enterprise Networks	5

This page intentionally left blank

Day 15

QoS

ENCOR 350-401 Exam Topics

Architecture

- Describe concepts of wired and wireless QoS
    - QoS components
    - QoS policy

Key Topics

Today we review concepts and mechanisms related to quality of service (QoS). As user applications continue to drive network growth and evolution, the demand to support various types of traffic is also increasing. Network traffic from business-critical and delay-sensitive applications must be serviced with priority and protected from other types of traffic. QoS, a crucial element of any administrative policy, mandates how to handle application traffic on a network. QoS and its implementations in a converged network are complex and create many challenges for network administrators and architects. Many QoS building blocks or features operate in different parts of a network to create an end-to-end QoS system. Managing how these building blocks are assembled and how different QoS features are used is critical for prompt and accurate delivery of data in an enterprise network.

Quality of Service

Networks must provide secure, predictable, measurable, and guaranteed services. End users want their applications to perform correctly, with no voice call drops; smooth, high-quality video; and rapid response time for data applications. However, different types of traffic that modern converged networks carry have very different requirements in terms of bandwidth, delay, jitter (delay variation), and packet loss. If these requirements are not met, the quality of applications may be degraded, and users will have reason to complain.

Need for Quality of Service

QoS is the ability of a network to predictably provide business applications with the service required for those applications to be successfully used on the network. The fundamental purpose of QoS is to manage contention for network resources in order to maximize the end-user experience of a session.

The goal of QoS is to provide better and more predictable network service via dedicated bandwidth, controlled jitter and latency, and improved loss characteristics, as required by business applications. QoS achieves these goals by providing tools that manage network congestion, shape network traffic, use expensive wide-area links more efficiently, and set traffic policies across the network.

QoS is not a substitute for bandwidth. If a network is congested, packets will be dropped. QoS allows administrators to control how, when, and what traffic is dropped during congestion. With QoS, when there is contention on a link, less important traffic is delayed or dropped in favor of delay-sensitive, business-important traffic.

QoS gives priority to some sessions over others. Packets of delay-sensitive sessions bypass queues of packets belonging to non-delay-sensitive sessions. When queue buffers overflow, either packets are dropped on sessions that can recover from the loss or those sessions can be eliminated with minimal business impact.

Converged Networks

Converged networks carry multiple types of traffic, such as voice, video, and data, that were traditionally transported on separate and dedicated networks. Although there are several advantages to converged networks, merging these different traffic streams, which have dramatically different requirements, can lead to several quality problems.

Voice and video are not tolerant of delay, jitter, or packet loss, and excessive amounts of any of these issues will result in a poor experience for end users. Data flows are typically more tolerant of delay, jitter, and packet loss, but they are very bursty in nature and typically use as much bandwidth as possible and as available.

The different traffic flows on a converged network are in competition for network resources. Unless some mechanism mediates the overall traffic flow, voice and video quality will be severely compromised at times of network congestion. The critical, time-sensitive flows must be given priority to preserve the quality of this traffic.

Multimedia streams, such as those used in IP telephony or video conferencing, are sensitive to delivery delays. Excessive delay can cause noticeable echo or talker overlap. Voice transmissions can be choppy or unintelligible, with high packet loss or jitter. Images may be jerky, or the sound might not be synchronized with the image. Voice and video calls may disconnect or may not connect at all if signaling packets are not delivered.

QoS can also severely affect some data applications. Time-sensitive applications, such as virtual desktop or interactive data sessions, may appear unresponsive. Delayed application data could have serious performance implications for users who depend on timely responses, such as in brokerage houses or call centers.

Four major problems affect quality on converged networks:

- **Bandwidth capacity:** Large graphic files, multimedia, and voice and video can cause bandwidth capacity problems over data networks. Multiple traffic flows compete for a limited amount of bandwidth and may require more bandwidth than is available.
- **Delay:** Delay is the time it takes for a packet to reach the receiving endpoint after the sender transmits the packet. This period, called the end-to-end delay, consists of variable-delay

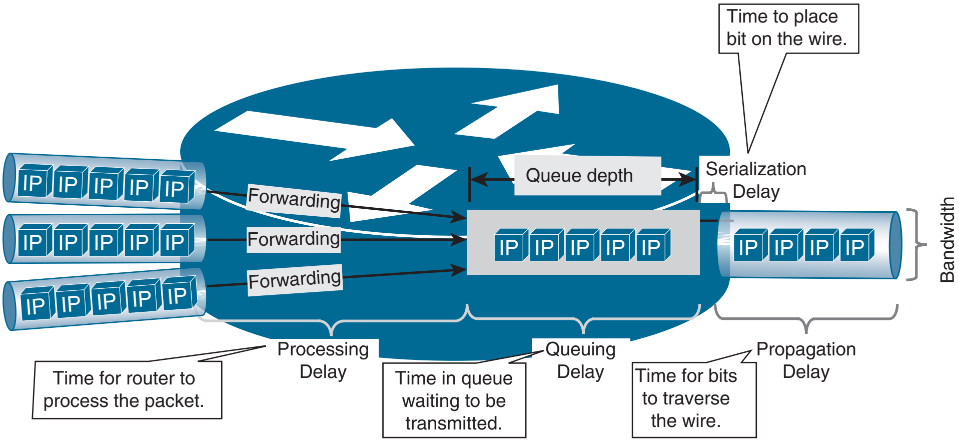
components (processing and queuing delay) and fixed-delay components (serialization and propagation delay). End-to-end delay is also referred to as *network latency*.

- **Jitter:** Jitter is the variation in end-to-end delay that is experienced between packets in the same flow as they traverse the network. This delta in end-to-end delay for any two packets is the result of the variable network delay.
- **Packet loss:** Congestion, faulty connectivity, and faulty network equipment are the usual causes of lost packets.

Components of Network Delay

There are four types of network delay, as shown in Figure 15-1:

Figure 15-1 Types of Network Delay


Time to place bit on the wire.
IP
Forwarding
Forwarding
Forwarding
Queue depth
Serialization Delay
Bandwidth
Processing Delay
Queuing Delay
Propagation Delay
Time for router to process the packet.
Time in queue waiting to be transmitted.
Time for bits to traverse the wire.

- **Processing delay (variable):** This delay is the time it takes for a router to move a packet from an input interface to the output queue of the output interface. The processing delay can vary and depends on these factors:
  - CPU speed
  - CPU utilization
  - IP switching mode
  - Router architecture
  - Configured features on both input and output interfaces, such as encryption and decryption, fragmentation and defragmentation, and address translation
- **Queuing delay (variable):** This delay is the time a packet resides in the output queue of a router. Queuing delay is variable and depends on the number and sizes of packets that are already in the queue, the bandwidth of the interface, and the queuing mechanism.
- **Serialization delay (fixed):** This delay is the time it takes to place a frame on the physical medium for transport. The serialization delay is a fixed value that is directly related to link bandwidth.

- **Propagation delay (fixed):** This delay is the fixed amount of time it takes to transmit a packet across a link and depends on the type of media interface and the link distance.

Variable-delay components can change based on conditions in the network—even for packets of the same size. Fixed-delay components increase linearly as packet size increases, but they remain constant for packets of the same size.

Delay can be managed by upgrading the link bandwidth, using a queuing technique to prioritize critical traffic, or enabling a compression technique to reduce the number of bits transmitted for packets on the link.

End-to-end network delay is calculated by adding all the network delay components along a given network path.

Jitter

Jitter is defined as a variation in the arrival (delay) of received packets. On the sending side, packets are sent in a continuous stream, and the packets are spaced evenly. Variable processing and queuing delays on network devices can cause this steady stream of packets to become uneven.

Congestion in the IP network is the usual cause of jitter. The congestion can occur at the router interfaces or in a provider or carrier network if the circuit has not been provisioned correctly. However, there can also be other sources of jitter:

- **Encapsulation:** The easiest and best place to start looking for jitter is at the router interfaces because you have direct control over this portion of the circuit. How you track down the source of jitter depends greatly on the encapsulation and type of link where the jitter happens. For example, in Point-to-Point Protocol (PPP) encapsulation, jitter is almost always due to serialization delay, which can easily be managed with link fragmentation and interleaving (LFI) on the PPP link. PPP endpoints talk directly to each other, without a network of switches between them, which means you have control over all the interfaces involved.

- **Fragmentation:** Fragmentation is more commonly associated with serialization delay than with jitter. However, under certain conditions, it can be the cause of jitter. If you incorrectly configure LFI on a slow-speed link, your media packets may become fragmented and thus increase jitter.

When there is excessive jitter for media traffic in a network, you may experience a choppy or synthetic-sounding voice. A choppy voice includes gaps in which syllables appear to be dropped or badly delayed in a start-and-stop fashion. A synthetic-sounding voice has an artificial quality and may also have a quiver or fuzziness. Predictive insertion causes this synthetic sound by replacing the sound that is lost when a packet is dropped with a best guess from a previous sample.

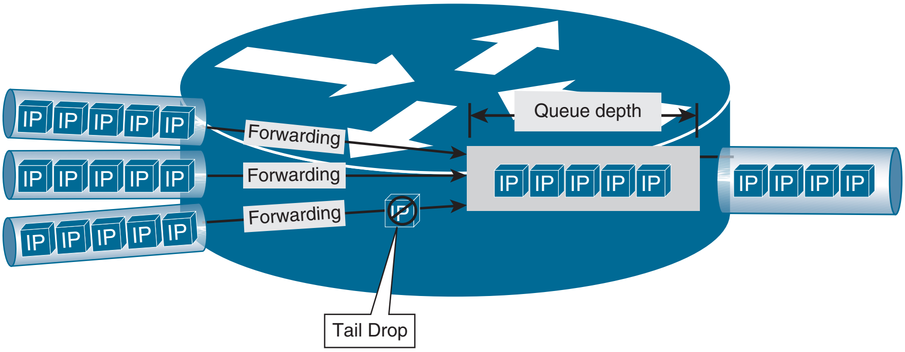
Dejitter Buffer Operation

When a media endpoint such as an IP phone or a video endpoint receives a stream of IP packets, it must compensate for the jitter encountered on the IP network. The mechanism that manages this function is a dejitter buffer, which is a time buffer. The dejitter buffer is provided by the terminating device to make the playout mechanism more effective. When a call starts, the dejitter buffer fills up. If media packets arrive too quickly, the queue fills; if media packets arrive too slowly, the queue empties. If a media packet is delayed beyond the holding capacity of the dejitter buffer, the packet is immediately dropped. If the packet is within the buffering capability, it is placed in the dejitter buffer. If the jitter is so significant that it causes packets to be received out of the range of this buffer, the out-of-range packets are discarded, and dropouts are heard in the audio.

Packet Loss

Packet loss typically occurs when routers run out of space for a particular interface output queue. Figure 15-2 illustrates a full interface output queue, which causes newly arriving packets to be dropped. Such drops are referred to as *output drops* or *tail drop*s (as packets are dropped at the tail of the queue).

Figure 15-2 Packet Loss



Routers might also drop packets for other less common reasons, including the following:

- **Input queue drop:** The main CPU is congested and cannot process packets (that is, the input queue is full).
- **Ignore:** The router ran out of buffer space.
- **Overrun:** The CPU is congested and cannot assign a free buffer to the new packet.
- **Frame errors:** There is a hardware-detected error in a frame, such as a CRC, runt, or giant.

Packet loss due to tail drop can be managed by increasing the link bandwidth, using a queuing technique that guarantees bandwidth and buffer space for applications that are sensitive to packet loss, or preventing congestion by shaping or dropping packets before congestion occurs. These solutions are discussed in the next section.

QoS Models

There are three models for implementing QoS on a network:

- **Best-effort model:** In a best-effort model, QoS is not applied to traffic. Packets are serviced in the order in which they are received, with no preferential treatment. The best-effort model is appropriate if it is not important when or how packets arrive and if there is no need to differentiate between traffic flows.

- **Integrated Services (IntServ) model:** The IntServ model provides guaranteed QoS to IP packets. Applications signal to the network that they will require special QoS for a period of time, and the appropriate bandwidth is reserved across the network. With IntServ, packet delivery is guaranteed; however, the use of this model can limit the scalability of the network.
- **Differentiated Services (DiffServ) model:** The DiffServ model provides scalability and flexibility in implementing QoS in a network. Network devices recognize traffic classes and provide different levels of QoS to different traffic classes.

Best-Effort QoS Model

If QoS policies are not implemented, traffic is forwarded using the best-effort model. All network packets are treated the same: An emergency voice message is treated exactly like a digital photograph that is attached to an email. Without QoS, the network cannot tell the difference between packets and, as a result, cannot treat packets preferentially.

When you drop a letter in standard postal mail, you are using a best-effort model. Your letter will be treated the same as every other letter. With the best-effort model, the letter may actually never arrive, and unless you have a separate notification arrangement with the letter recipient, you may never know if the letter arrives.

IntServ Model

Some applications, such as high-definition video conferencing, require consistent, dedicated bandwidth to provide a sufficient experience for users. IntServ was introduced to guarantee predictable network behavior for these types of applications. Because IntServ reserves bandwidth throughout a network, no other traffic can use the reserved bandwidth.

IntServ provides hard end-to-end QoS guarantees, such as bandwidth, delay, and packet loss rates. These guarantees ensure predictable and guaranteed service levels for applications. There is no effect on traffic when guarantees are made because QoS requirements are negotiated when the connection is established. These guarantees require an end-to-end QoS approach with complexity and scalability limitations.

Using IntServ is like having a private courier airplane or truck that is dedicated to delivering your traffic. This model ensures quality and delivery, but it is expensive and is subject to scalability issues because it requires reserved resources that are not shared.

The IntServ solution allows end stations to explicitly request specific network resources. Resource Reservation Protocol (RSVP) provides a mechanism for requesting the network resources. If resources are available, RSVP accepts a reservation and installs a traffic classifier in the QoS forwarding path. The traffic classifier tells the QoS forwarding path how to classify packets from a particular flow and which forwarding treatment to provide. The IntServ standard assumes that routers along a path set and maintain the state for each individual communication.

DiffServ Model

DiffServ was designed to overcome the limitations of the best-effort and IntServ models. DiffServ can provide “almost guaranteed” QoS and is cost-effective and scalable.

With the DiffServ model, QoS mechanisms are used without prior signaling, and QoS characteristics (for example, bandwidth, delay) are managed on a hop-by-hop basis with policies that are established independently at each device in the network. This approach is not considered an end-to-end QoS strategy because end-to-end guarantees cannot be enforced. However, DiffServ is a more scalable approach to implementing QoS because hundreds or potentially thousands of applications can be mapped into a small set of classes on which similar sets of QoS behaviors are implemented. Although QoS mechanisms in this approach are enforced and applied on a hop-by-hop basis, uniformly applying global meaning to each traffic class provides flexibility and scalability.

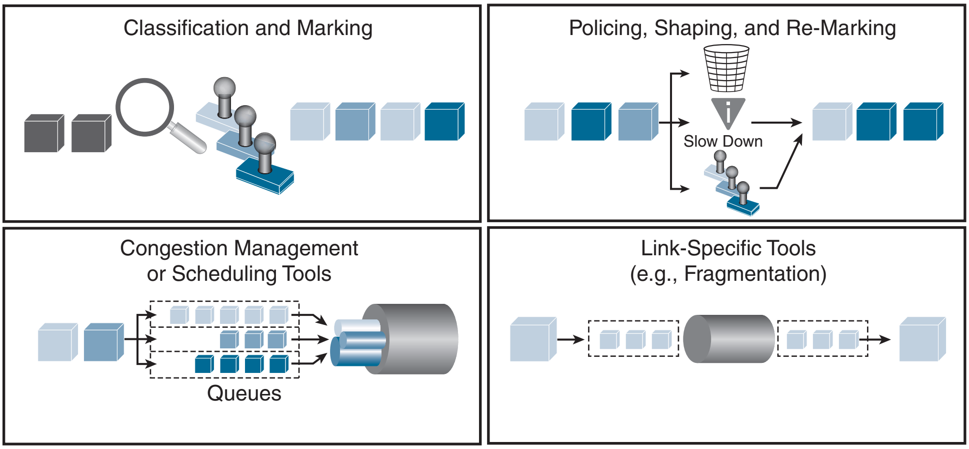
With DiffServ, network traffic is divided into classes that are based on business requirements. Each of the classes can then be assigned a different level of service. As the packets traverse a network, each of the network devices identifies the packet class and manages the packet according to this class. You can choose many levels of service with DiffServ. For example, voice traffic from IP phones and traffic from video endpoints are usually given preferential treatment over all other application traffic. Email is generally given best-effort service. Nonbusiness, or scavenger, traffic can be given very poor service or may be blocked entirely.

DiffServ works like a package delivery service. You request (and pay for) a level of service when you send your package. Throughout the package network, the level of service is recognized, and your package is given preferential or normal service, depending on your request.

QoS Mechanisms Overview

Generally, you can place QoS tools into the following four categories, as illustrated in Figure 15-3:

Figure 15-3 QoS Mechanisms



- **Classification and marking tools:** These tools analyze sessions to determine which traffic class they belong to and therefore which treatment the packets in the session should receive. Classification should happen as few times as possible because it takes time and uses up resources. For that reason, packets are marked after classification, usually at the ingress edge of a network. A packet might travel across different networks to its destination. Reclassification and re-marking are common at the hand-off points upon entry to a new network.

- **Policing, shaping, and re-marking tools:** These tools assign different classes of traffic to certain portions of network resources. When traffic exceeds available resources, some traffic might be dropped, delayed, or re-marked to avoid congestion on a link. Each session is monitored to ensure that it does not use more than the allotted bandwidth. If a session uses more than the allotted bandwidth, traffic is dropped (policing), slowed down (shaped), or re-marked (marked down) to conform.
- **Congestion management or scheduling tools:** When traffic exceeds the network resources that are available, traffic is queued. Queued traffic waits for available resources. Traffic classes that do not handle delay well are better off being dropped unless there is guaranteed delay-free bandwidth for the traffic class.
- **Link-specific tools:** Certain types of connections, such as WAN links, can be provisioned with special traffic-handling tools. One such example is fragmentation.

Classification and Marking

In any network in which networked applications require differentiated levels of service, traffic must be sorted into different classes on which quality of service (QoS) is applied. Classification and marking are two critical functions of any successful QoS implementation. Classification, which can occur from Layer 2 to Layer 7, allows network devices to identify traffic as belonging to a specific class with specific QoS requirements, as determined by an administrative QoS policy. After network traffic is sorted, individual packets are colored or marked so that other network devices can apply QoS features uniformly to those packets, in compliance with the defined QoS policy.

Classification

Classification involves identifying and sorting packets into different traffic types so that different policies can then be applied. Packet classification usually involves using a traffic descriptor to categorize packets within specific groups to define the packets. Classification of packets can happen without marking.

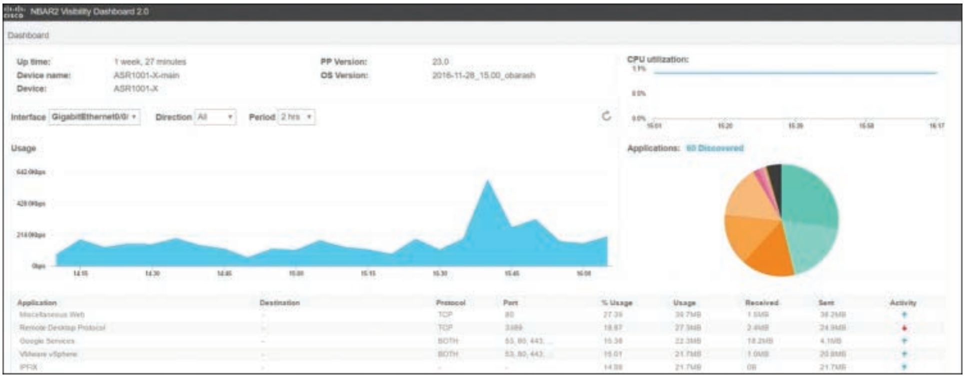
Classification involves inspecting one or more fields in a packet to identify the type of traffic the packet is carrying. After the packet has been defined (classified), the packet is then accessible for QoS handling on the network. Commonly used traffic descriptors include class of service (CoS), incoming interface, IP precedence, Differentiated Services Code Point (DSCP), source or destination address, application, and MPLS EXP bits.

Using packet classification, you can partition network traffic into multiple priority levels or classes of service. When traffic descriptors are used to classify traffic, the source agrees to adhere to the contracted terms, and the network promises a QoS level. Different QoS mechanisms, such as traffic policing, traffic shaping, and queuing techniques, use the traffic descriptor of the packet (the classification of the packet) to ensure adherence to this agreement.

NBAR

Network Based Application Recognition (NBAR) is a feature in Cisco IOS Software that provides intelligent classification for the network infrastructure. Cisco NBAR is a classification engine that can recognize a wide variety of protocols and applications, including web-based applications and client and server applications that dynamically assign TCP or UDP port numbers. After a protocol or an application is recognized, the network can invoke specific services for that particular protocol or application. Figure 15-4 shows the NBAR2 HTTP-based Visibility Dashboard, which provides a graphical display of network information, such as network traffic details and bandwidth utilization. The Visibility Dashboard includes interactive charts and a graph of bandwidth usage for detected applications.

Figure 15-4 NBAR2 Visibility Dashboard



Up time: 1 week, 27 minutes
Device name: ASR1001-X-main
Device: ASR1001-X
PP Version: 23.0
OS Version: 2016-11-28_15.00_obarash
Interface GigabitEthernet0/0/
Direction All
Period 2 hrs

Application	Destination	Protocol	Port	% Usage	Usage	Received	Sent	Activity
Miscellaneous Web	-	TCP	80	27.39	39.7MB	1.5MB	38.2MB	↑
Remote Desktop Protocol	-	TCP	3389	18.87	27.3MB	2.4MB	24.9MB	↓
Google Services	-	BOTH	53, 80, 443, ...	15.38	22.3MB	18.2MB	4.1MB	↑
VMware vSphere	-	BOTH	53, 80, 443, ...	15.01	21.7MB	1.0MB	20.8MB	↑
IPFIX	-	-	-	14.98	21.7MB	0B	21.7MB	↑

Cisco NBAR can perform deep packet Layer 4 through Layer 7 inspection to identify applications, based on information in the packet payload, and can perform stateful bidirectional inspection of traffic as it flows through the network.

When used in active mode, Cisco NBAR is enabled in the Modular QoS (MQC) structure as a mechanism to classify traffic. For Cisco NBAR, the criteria for classifying packets into class maps are whether the packet matches a specific protocol or application that is known to NBAR. Using the MQC, network traffic with one network protocol (for example, Citrix) can be placed into one traffic class, and traffic that matches a different network protocol (for example, Skype) can be placed into another traffic class. You can then set different Layer 3 marking values to different classes of traffic.

When used in passive mode, NBAR Protocol Discovery is enabled on a per-interface basis to discover and provide real-time statistics on applications.

Next-generation NBAR, or NBAR2, is a fully backward-compatible re-architecture of Cisco NBAR with advanced classification techniques, accuracy, and more signatures. NBAR2 is supported on multiple devices, including the Generation 2 Cisco Integrated Services Router (ISR), the Cisco 1000 Aggregation Services Router (ASR), the ISR 4400, the Cisco 1000 Series Cloud Services Router (CSR), the Cisco Adaptive Security Appliance with Context-Aware Security (ASA-CX), and Cisco wireless LAN controllers (WLCs).

Cisco NBAR protocol and signature support can be updated by installing the Packet Description Language Module (PDLM) for NBAR systems or protocol packs for NBAR2 systems. This support allows for nondisruptive updates to the NBAR capabilities by not requiring an update from the base image.

Example 15-1 shows some of the application matching options that NBAR2 offers and that can be used to apply different QoS policies to different traffic streams.

Example 15-1 Configuring NBAR2 Application Matching

```
Router(config-cmap)# match protocol attribute category ?
  anonymizers                      Anonymizers applications
  backup-and-storage               Backup and storage related applications
  browsing                         Browsing related applications
  business-and-productivity-tools  Business-and-productivity-tools related
                                   applications
  database                         Database related applications
  email                            Email related applications
  epayement                        Epayement related applications
  file-sharing                     File-sharing related applications
  gaming                           Gaming related applications
  industrial-protocols             Industrial-protocols related applications
  instant-messaging                Instant-messaging related applications
  inter-process-rpc                Inter-process-rpc related applications
  internet-security                Internet security related applications
  layer3-over-ip                   Layer3-over-IP related applications
  location-based-services          Location-based-services related applications
  net-admin                        Net-admin related applications
  newsgroup                        Newsgroup related applications
  other                            Other related applications
```

Marking

Marking involves assigning different identifying values (traffic descriptors) to headers of an incoming packet or frame. Marking, which is related to classification, allows network devices to classify a packet or frame using a specific traffic descriptor that was previously applied to it. Marking can be used to set information in the Layer 2 or Layer 3 packet headers.

When network traffic is coming to the network edge, it usually does not have any applied marking value, so you need to perform classification that is based on other parameters, such as IP addresses, TCP/UDP ports, or the protocol field. Some network devices can even check for application details, such as HTTP, MIME, or RTP payload type, to properly classify network traffic. This method of classification is considered complex because the network device must open each packet or frame in a traffic flow and look at its contents to properly classify it. However, marking a packet or frame allows network devices to easily distinguish a marked packet or frame as belonging to a specific class. So instead of performing complex classification, a network device can simply look at the packet or frame header and classify traffic based on the marking value that was previously assigned.

This approach allows a network device to save CPU and memory resources and makes QoS more efficient.

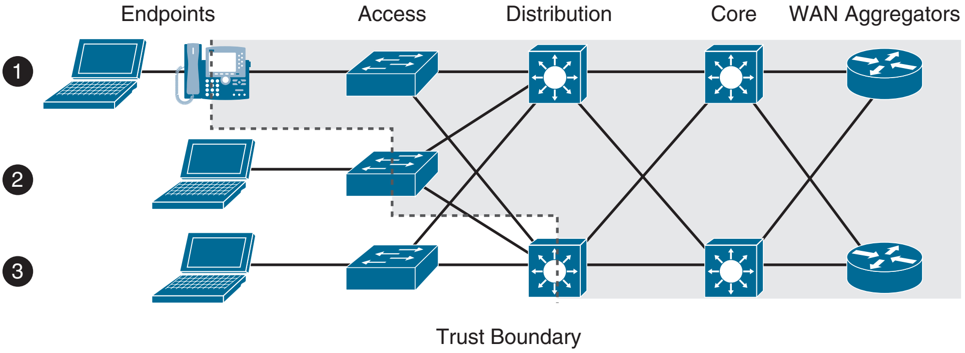
You should apply marking as close to the source of the traffic as possible, such as at the network edge, typically in the wiring closet. In this case, you perform complex classification only on the edge of the network, and none of the subsequent network devices have to repeat in-depth classification and analysis (which can be computationally intensive tasks) to determine how to treat a packet. After the packets or frames are identified as belonging to a specific class, other QoS mechanisms can use these markings to uniformly apply QoS policies.

The concept of trust is important in deploying QoS marking. When an end device (such as a workstation or an IP phone) marks a packet with CoS or DSCP, a switch or router can accept or reject values from the end device. If the switch or router chooses to accept the values, the switch or router trusts the end device. If the switch or router trusts the end device, it does not need to do any re-marking of packets from this interface. If the switch or router does not trust the interface, it must perform a reclassification to determine the appropriate QoS value to be assigned to the packets coming from this interface. Switches and routers are generally set to not trust end devices and must be specifically configured to trust packets coming from an interface.

The point where packet markings are not necessarily trusted is called the *trust boundary*. You can create, remove, or rewrite markings at that point. The borders of a trust domain are the network locations where packet markings are accepted and acted on. In an enterprise network, the trust boundary is typically at the access layer switch. Switch ports can be configured in untrusted state, trusted CoS state, or trusted DSCP state.

Figure 15-5 illustrates the optimal location for the trust boundary (line 1 and line 2). When a trusted endpoint is connected to the access layer switch, the trust boundary can be extended to it (as illustrated in line 1). When an untrusted endpoint is connected to the switch, the trust boundary ends at the access layer (as illustrated in line 2). Finally, line 3 of the diagram shows the suboptimal placement of the trust boundary at the distribution layer.

Figure 15-5 QoS Trust Boundary



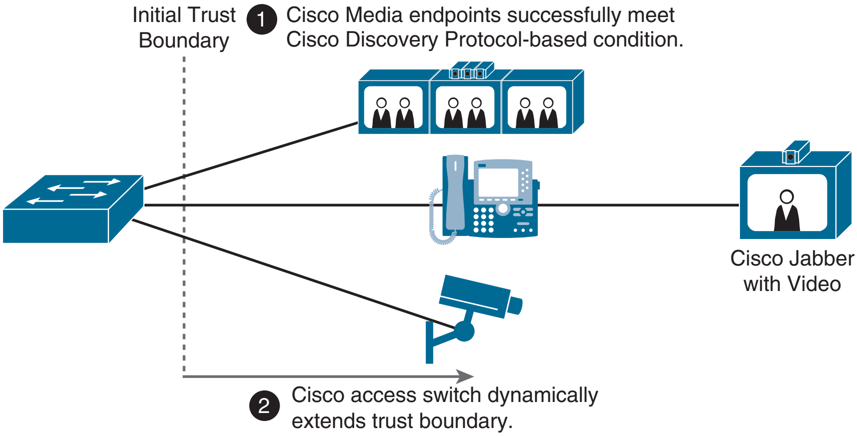
In order to understand the operation of various trust states, you need to know about the three static states to which a switch port can be configured:

- **Untrust:** In this state, the port discards any Layer 2 or Layer 3 markings and generates an internal Differentiated Services Code Points (DSCP) value of 0 for the packet.

- **Trust CoS:** In this state, the port accepts class of service (CoS) marking and calculates the internal DSCP value according to the default or predefined CoS-to-DSCP mapping.
- **Trust DSCP:** In this state, the port trusts the DSCP marking, and it sets the internal DSCP value to match the received DSCP value.

Besides static configuration of trust, Cisco Catalyst switches can also define a dynamic trust state so that trust on a port dynamically depends on endpoint identification according to the trust policy. Such endpoint identification depends on Cisco Discovery Protocol, and it is therefore supported for Cisco end devices only. Figure 15-6 illustrates this concept. When the switch receives CDP messages, the trust boundary is extended to the Cisco devices, and their QoS markings are trusted.

Figure 15-6 QoS Dynamic Trust Boundary


Initial Trust Boundary
①
Cisco Media endpoints successfully meet Cisco Discovery Protocol-based condition.
Cisco Jabber with Video
②
Cisco access switch dynamically extends trust boundary.

Layer 2 Classification and Marking

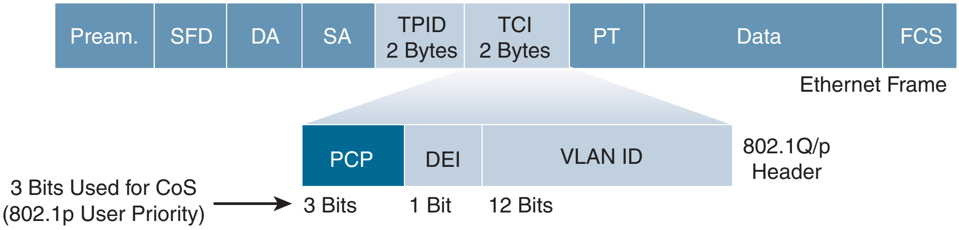
The packet classification and marking options that are available at the data link layer depend on the Layer 2 technology. At the network layer, IP packets are commonly classified based on source or destination IP address, packet length, or the contents of the type of service (ToS) byte.

Each data link technology has its own mechanism for classification and marking. Each technique is meaningful only to this Layer 2 technology and is bound by the extent of the Layer 2 network. For the marking to persist beyond the Layer 2 network, translation of the relevant field must take place.

802.1p Class of Service

The 802.1Q standard is an Institute of Electrical and Electronics Engineers (IEEE) specification for implementing VLANs in Layer 2 switched networks. The 802.1Q specification defines two 2-byte fields, Tag Protocol Identifier (TPID) and Tag Control Information (TCI), which are inserted in an Ethernet frame following the source address field. The TPID field is fixed and assigned the value 0x8100. The TCI field is composed of three fields, as illustrated in Figure 15-7:

Figure 15-7 QoS Class of Service



- **PCP (3 bits):** The IEEE 802.1p standard defines the specifications of the 3-bit Priority Code Point (PCP) field. These bits can be used to mark packets as belonging to a specific CoS. The CoS markings use the three 802.1p user priority bits and allow a Layer 2 Ethernet frame to be marked with eight levels of priority (values 0 through 7). The 3 bits allow a direct correspondence with IPv4 (IP precedence) type of service (ToS) values. The 802.1p specification defines these standard definitions for each CoS, as shown in Table 15-1.

Table 15-1 IEEE 802.1p CoS

PCP Value / Priority	Traffic Type
1 (lowest)	Background (BK)
0 (default)	Best effort (BE)
2	Excellent effort (EE)
3	Critical applications (CA)
4	Video (VI)
5	Voice (VO)
6	Internetwork control (IC)
7 (highest)	Network control (NC)

The default priority for transmission by end stations is 0. Changing this default would result in confusion and probably also interoperability problems. At the same time, the default traffic type is Best Effort; 0 is thus used both for default priority and for Best Effort, and Background is associated with a priority value of 1. This means that the value 1 effectively communicates a lower priority than 0.

One disadvantage of using CoS marking is that frames lose their CoS markings when transiting a non-802.1Q or non-802.1p link, including any type of non-Ethernet WAN link. Therefore, a more permanent marking should be used for network transit, such as Layer 3 IP DSCP marking. This goal is typically accomplished by translating a CoS marking into another marker or simply using a different marking mechanism.

- **DEI (1 bit):** This bit indicates frames eligible to be dropped in the presence of congestion. This bit can be used in conjunction with the PCP field.

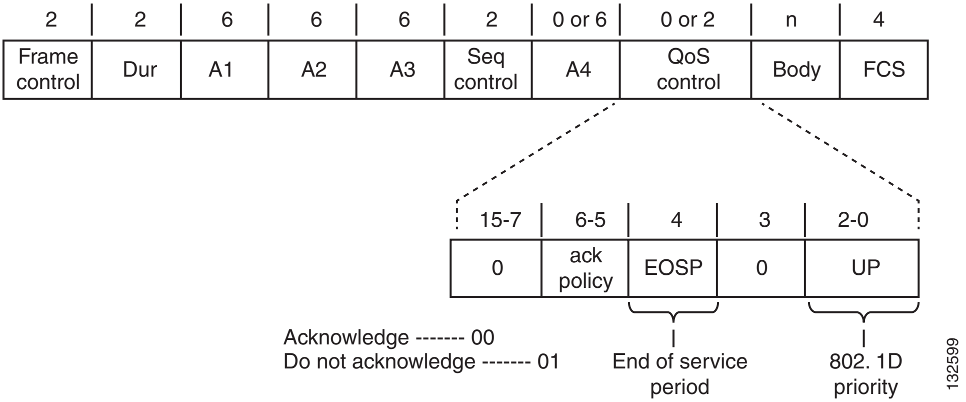
- **VLAN ID (12 bits):** The VLAN ID field is a 12-bit field that defines the VLAN that is used by 802.1Q. Because this field is 12 bits, the number of VLANs supported by 802.1Q is restricted to 4096.

802.11 Wireless QoS: 802.11e

Wireless access points are the second-most-likely places in the enterprise network to experience congestion (after LAN-to-WAN links). This is because wireless media generally presents a downshift in speed/throughput, it is half-duplex, and it is shared. The case for QoS on the WLAN is to minimize packet drops due to congestion, as well as minimize jitter due to nondeterministic access to the half-duplex, shared media.

The IEEE 802.11e standard includes, among other QoS features, user priorities and access categories, as well as clear UP-to-DSCP mappings. 802.11e introduced a 3-bit marking value in Layer 2 wireless frames referred to as User Priority (UP). UP values range from 0 to 7. Figure 15-8 shows the UP field in the QoS Control field of the 802.11 MAC header.

Figure 15-8 802.11 Wi-Fi MAC Frame QoS Control Field


2
2
6
6
6
2
0 or 6
0 or 2
n
4
Frame control
Dur
A1
A2
A3
Seq control
A4
QoS control
Body
FCS
15-7
6-5
4
3
2-0
0
ack policy
EOSP
0
UP
Acknowledge ------- 00
Do not acknowledge ------- 01
End of service period
802. 1D priority
132599

Pairs of UP values are assigned to four access categories (AC), which statistically equate to four distinct levels of service over the WLAN. Table 15-2 lists access categories and their UP pairings.

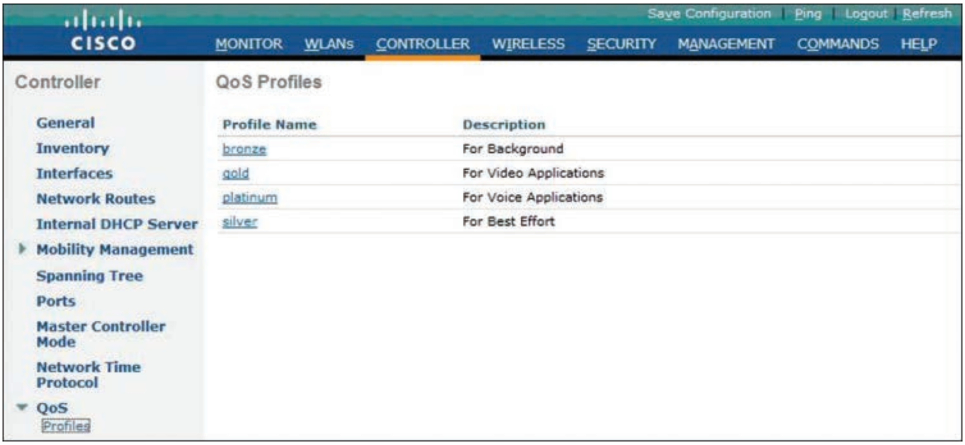
Table 15-2 IEEE 802.1e Access Categories

802.11e UP Value (802.1p)	802.11e Access Category	WMM Designation	Cisco WLC QoS Profile Name
7 (Network control)	AC_VO	Voice	Platinum
6 (Voice)			
5 (Video)	AC_VI	Video	Gold
4 (Controlled load)			
3 (Excellent effort)	AC_BE	Best effort	Silver
0 (Best effort)			
2 (Spare) 1 (Background)	AC_BK	Background	Bronze

Table 15-2 demonstrates how the four wireless ACs map to their corresponding 802.11e/WMM UP values. For reference, this table also shows the corresponding names of these ACs that are used in the Cisco WLCs. Instead of using the normal WMM naming convention for the four ACs, Cisco uses a precious metals naming system, but a direct correlation exists to these four ACs.

Figure 15-9 shows the four QoS profiles that can be configured on a Cisco WLAN controller: platinum, gold, silver, and bronze.

Figure 15-9 Cisco WLC QoS Profiles


Save Configuration
Ping
Logout
Refresh
cisco
MONITOR
WLANs
CONTROLLER
WIRELESS
SECURITY
MANAGEMENT
COMMANDS
HELP
Controller
General
Inventory
Interfaces
Network Routes
Internal DHCP Server
Mobility Management
Spanning Tree
Ports
Master Controller Mode
Network Time Protocol
QoS
Profiles

QoS Profiles

Profile Name	Description
bronze	For Background
gold	For Video Applications
platinum	For Voice Applications
silver	For Best Effort

Layer 3 Marking: IP Type of Service

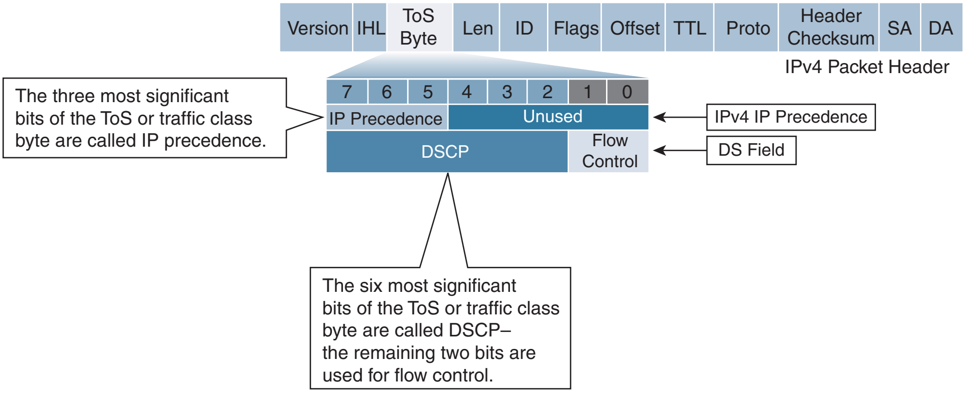
At Layer 3, IP packets are commonly classified based on the source or destination IP address, packet length, or contents of the ToS byte. Classification and marking in IP packets occur in the ToS byte for IPv4 packets and occur in the traffic class byte for IPv6 packets.

Link layer media often change as a packet travels from its source to its destination. Because a CoS field does not exist in a standard Ethernet frame, CoS markings at the link layer are not preserved as packets traverse nontrunked or non-Ethernet networks. Using marking at Layer 3 provides a more permanent marker that is preserved from the source to the destination.

Originally, only the first 3 bits of the ToS byte (known as the IP Precedence bits) were used for marking. However, newer standards use the first 6 bits of the ToS byte (known as the DSCP bits) for marking.

The header of an IPv4 packet contains the ToS byte. The 3 IP Precedence bits in the ToS field of the IPv4 header specify the service class for each packet. IP Precedence values range from 0 to 7 and allow you to partition traffic in up to six usable classes of service. Settings 6 and 7 are reserved for internal network use. Figure 15-10 shows both the IP Precedence and DSCP bits in the ToS byte.

Figure 15-10 QoS Type of Service


Version
IHL
ToS Byte
Len
ID
Flags
Offset
TTL
Proto
Header Checksum
SA
DA
IPv4 Packet Header
7
6
5
4
3
2
1
0
IP Precedence
Unused
DSCP
Flow Control
IPv4 IP Precedence
DS Field
The three most significant bits of the ToS or traffic class byte are called IP precedence.
The six most significant bits of the ToS or traffic class byte are called DSCP– the remaining two bits are used for flow control.

The DiffServ model supersedes—and is backward compatible with—IP precedence. DiffServ redefines the ToS byte as the DiffServ field and uses 6 prioritization bits that permit classification of up to 64 values (0 to 63), of which 32 are commonly used. A DiffServ value is commonly referred to as a DSCP value.

With DiffServ, packet classification is used to categorize network traffic into multiple priority levels or classes of service. Packet classification uses the DSCP traffic descriptor to categorize a packet within a specific group to define this packet. After the packet has been defined (classified), the packet is then accessible for QoS handling on the network.

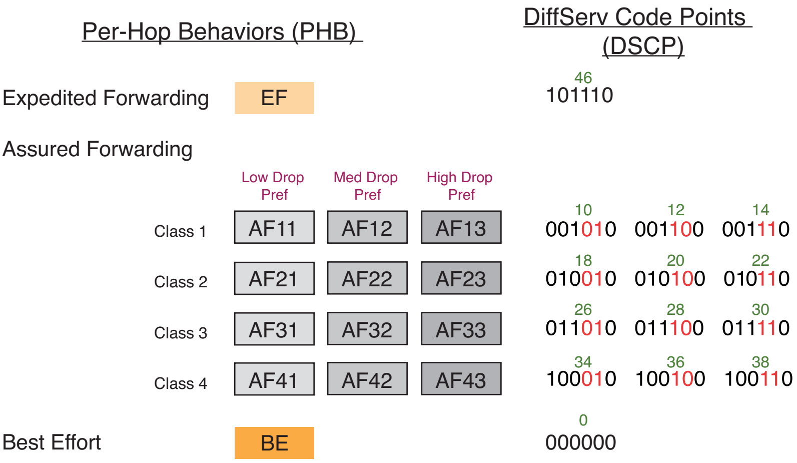
The last 2 bits of the ToS byte (the Flow Control bits) are reserved for explicit congestion notification (ECN), which allows end-to-end notification of network congestion without dropping packets. ECN is an optional feature that may be used between two ECN-enabled endpoints when the underlying network infrastructure also supports it.

When ECN is successfully negotiated, an ECN-aware router may set a mark in the IP header instead of dropping a packet to signal impending congestion. The receiver of the packet echoes the congestion indication to the sender, which reduces its transmission rate as though it detected a dropped packet. Because ECN marking in routers depends on some form of active queue management, routers must be configured with a suitable queue discipline to perform ECN marking. Cisco IOS routers perform ECN marking if configured with the weighted random early detection (WRED) queuing discipline.

Layer 3 Marking: DSCP Per-Hop Behaviors

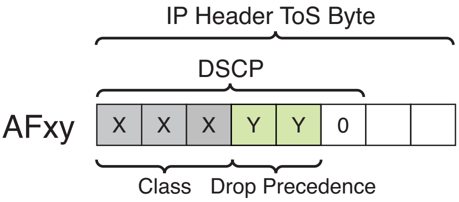
The 6-bit DSCP fields used in IPv4 and IPv6 headers are encoded as shown in Figure 15-11. DSCP values can be expressed in numeric form or by using special keyword names, called per-hop behaviors (PHBs). Three defined classes of DSCP PHBs exist: Best-Effort (BE or DSCP 0), Assured Forwarding (AF*xy*), and Expedited Forwarding (EF). In addition to these three defined PHBs, Class-Selector (CSx) code points have been defined to be backward compatible with IP precedence. (In other words, CS1 through CS7 are identical to IP precedence values 1 through 7.) The RFCs describing these PHBs are RFCs 2547, 2597, and 3246.

Figure 15-11 DSCP Encoding Scheme


Per-Hop Behaviors (PHB)
DiffServ Code Points (DSCP)
Expedited Forwarding
EF
46
101110
Assured Forwarding
Low Drop Pref
Med Drop Pref
High Drop Pref
Class 1
AF11
AF12
AF13
10
001010
12
001100
14
001110
Class 2
AF21
AF22
AF23
18
010010
20
010100
22
010110
Class 3
AF31
AF32
AF33
26
011010
28
011100
30
011110
Class 4
AF41
AF42
AF43
34
100010
36
100100
38
100110
Best Effort
BE
0
000000

RFC 2597 defines four Assured Forwarding classes, denoted by the letters AF followed by two digits. The first digit denotes the AF class and can range from 1 through 4. The second digit refers to the level of drop preference within each AF class and can range from 1 (lowest drop preference) to 3 (highest drop preference). For example, during periods of congestion (on an RFC 2597–compliant node), AF33 would statistically be dropped more often than AF32, which, in turn, would be dropped more often than AF31. Figure 15-12 shows the AF PHB encoding scheme.

Figure 15-12 DSCP Assured Forwarding Encoding Scheme


IP Header ToS Byte
DSCP
AFxy
X
X
X
Y
Y
0
Class
Drop Precedence

Mapping Layer 2 to Layer 3 Markings

Layer 2 CoS or Layer 3 IP precedence values generally constitute the 3 most significant bits of the equivalent 6-bit DSCP value and therefore map directly to the Code Selector (CS) points defined by the DiffServ RFCs.

For example, CoS 5 (binary 101) maps to DSCP 40 (binary 101000). Using the layout given in Figure 15-12, the mapping is formed by replacing the *XXX* value in the figure with the CoS value, and the *YY* value remains 0. Table 15-3 shows the mappings between CoS, CS, and DSCP values.

Table 15-3 Layer 2 CoS to Layer 3 Class Selector/DSCP Mappings

802.1p CoS Value	CoS Binary Equivalent	Class Selector	CS Binary Equivalent	DSCP Value (Decimal)
0	000	CS0/DF	000 000	0
1	001	CS1	001 000	8
2	010	CS2	010 000	16
3	011	CS3	011 000	24
4	100	CS4	100 000	32
5	101	CS5	101 000	40
6	110	CS6	110 000	48
7	111	CS7	111 000	56

Mapping Markings for Wireless Networks

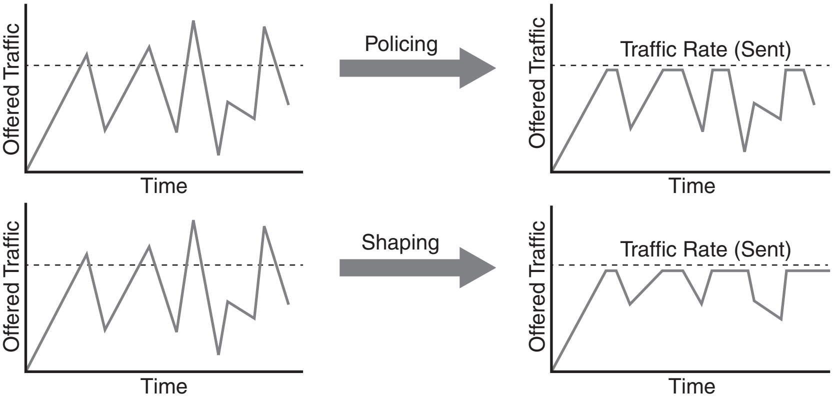
Cisco wireless products support Wi-Fi MultiMedia (WMM), a QoS system based on the IEEE 802.11e standard and published by the Wi-Fi Alliance. The IEEE 802.11 Wi-Fi classifications are different from how Cisco wireless technology deals with classification (based on IETF RFC 4594). The primary difference in classification is the changing of voice and video traffic to CoS 5 and 4, respectively (from 6 and 5, used by the IEEE 802.11 Wi-Fi). This makes it possible for the six classifications to be used for Layer 3 network control. To be compliant with both standards, the Cisco Unified Wireless Network solution performs conversions between the various classification standards when the traffic crosses the wireless-wired boundary.

Policing, Shaping, and Re-marking

After you identify and mark traffic, you can treat it by using a set of actions, including bandwidth assignment, policing, shaping, queuing, and dropping decisions.

Policers and shapers are tools that identify and respond to traffic violations. They usually identify traffic violations in a similar manner, but they differ in their response, as illustrated in Figure 15-13:

Figure 15-13 QoS Policing and Shaping Comparison



- **Policers:** Policers perform checks for traffic violations against a configured rate. The action they take in response is either dropping or re-marking the excess traffic. Policers do not delay traffic; they only check traffic and take action if needed.
- **Shapers:** Shapers are traffic-smoothing tools that work in cooperation with buffering mechanisms. A shaper does not drop traffic, but it smooths it out, so it never exceeds the configured rate. Shapers are usually used to meet SLAs. Whenever the traffic spikes above the contracted rate, the excess traffic is buffered and thus delayed until the offered traffic goes below the contracted rate.

Policers make instantaneous decisions and are thus optimally deployed as ingress tools. The logic is that if you are going to drop a packet, you might as well drop it before spending valuable bandwidth and CPU cycles on it. However, policers can also be deployed at egress to control the bandwidth that a particular class of traffic uses. Such decisions sometimes cannot be made until the packet reaches the egress interface.

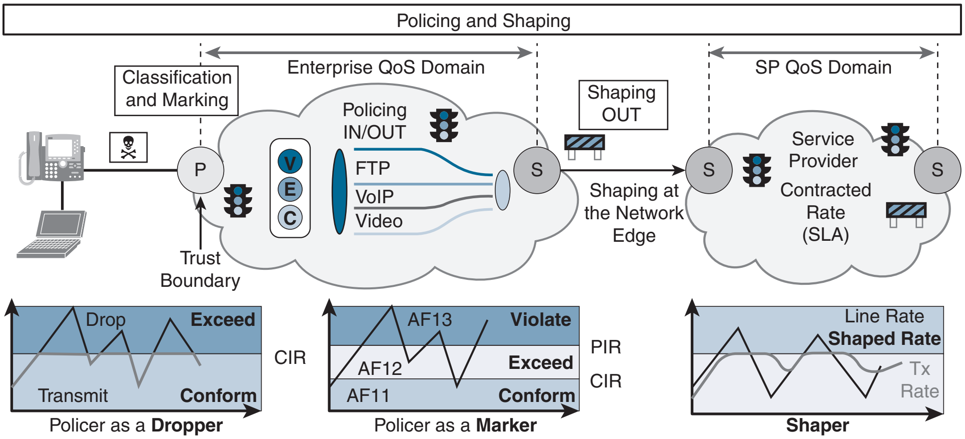
When traffic exceeds the allocated rate, the policer can take one of two actions: It can either drop traffic or re-mark it to another class of service. The new class usually has a higher drop probability.

Shapers are commonly deployed on enterprise-to-service provider links on the enterprise egress side. Shapers ensure that traffic going to the service provider does not exceed the contracted rate. If the traffic exceeds the contracted rate, it gets policed by the service provider and is likely to be dropped.

Policers can cause a significant number of TCP re-sends when traffic is dropped, but policing does not cause delay or jitter in a traffic stream. Shaping involves fewer TCP re-sends but does cause delays and jitter.

Figure 15-14 illustrates policing as user traffic enters the enterprise network and shaping as it exits. In the figure, CIR refers to the committed information rate, which is the rate, in bits per second, specified in the service-level agreement (SLA) with the service provider. The PIR is the peak information rate, which is the maximum rate of traffic allowed on the circuit.

Figure 15-14 QoS Policing and Shaping Across the Enterprise Network



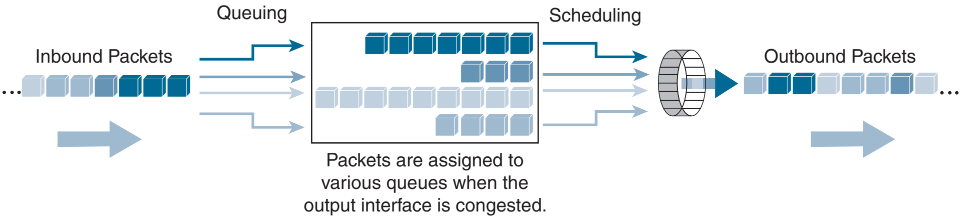
Policing and Shaping
Enterprise QoS Domain
SP QoS Domain
Classification and Marking
Policing IN/OUT
Shaping OUT
P
V
E
C
FTP
VoIP
Video
S
Shaping at the Network Edge
S
Service Provider
Contracted Rate (SLA)
S
Trust Boundary
Drop
Exceed
Transmit
Conform
CIR
Policer as a Dropper
AF13
Violate
AF12
Exceed
AF11
Conform
PIR
CIR
Policer as a Marker
Line Rate
Shaped Rate
Tx Rate
Shaper

Managing Congestion

Whenever a packet enters a device faster than it can exit, the potential for congestion occurs. If there is no congestion, packets are sent when they arrive. If congestion occurs, congestion management tools are activated. Queuing is temporary storage of backed-up packets. You perform queuing to avoid dropping packets.

Congestion management includes queuing (or buffering). It uses a logic that reorders packets into output buffers. It is only activated when congestion occurs. When queues fill up, packets can be reordered so that the higher-priority packets can be sent out of the exit interface sooner than the lower-priority packets. This is illustrated in Figure 15-15.

Figure 15-15 QoS Congestion Management



Scheduling is a process that involves deciding which packet should be sent out next. Scheduling occurs regardless of whether there is congestion on the link.

Low-latency queuing takes the previous model and adds a queue with strict priority (for real-time traffic).

Different scheduling mechanisms exist. The following are three basic examples:

- **Strict priority queuing:** The queues with lower priority are served only when the higher-priority queues are empty. There is a risk with this kind of scheduling that the lower-priority traffic will never be processed. This situation is commonly referred to as *traffic starvation*.
- **Round-robin queuing:** Packets in queues are served in a set sequence. There is no starvation with this scheduler, but delays can badly affect the real-time traffic.
- **Weighted fair queuing:** Queues are weighted so that some are served more frequently than others. This method thus prevents starvation and also gives priority to real-time traffic. One drawback of this method is that it does not provide bandwidth guarantees. The resulting bandwidth per flow instantaneously varies based on the number of flows present and the weights of each of the other flows.

The scheduling tools used for QoS deployments offer a combination of these algorithms and various ways to mitigate their downsides. By choosing a good combination, you can tune your network for the actual traffic flows that are present.

Class-Based Weighted Fair Queuing

A modern QoS example from Cisco is class-based weighted fair queuing (CBWFQ). With this type of queuing, traffic classes get fair bandwidth guarantees. There are no latency guarantees, so CBWFQ is only suitable for data networks.

There are many different queuing mechanisms. Older methods are insufficient for modern rich-media networks. However, you need to understand these older methods to comprehend the newer methods:

- **First-in, first-out (FIFO):** This method involves a single queue with packets that are sent in the exact order in which they arrived.
- **Priority queuing (PQ):** PQ involves a set of four queues that are served in strict priority order. By enforcing strict priority, the lower-priority queues are served only when the higher-priority queues are empty. This method can starve traffic in the lower-priority queues.
- **Custom queuing (CQ):** CQ involves a set of 16 queues with a round-robin scheduler. To prevent traffic starvation, CQ provides traffic guarantees. The drawback of this method is that it does not provide strict priority queueing for real-time traffic.
- **Weighted fair queuing (WFQ):** WFQ is an algorithm that divides the interface bandwidth by the number of flows, thus ensuring proper distribution of the bandwidth for all applications. This method provides good service for the real-time traffic, but there are no guarantees for a particular flow.

Here are two examples of newer queuing mechanisms that are recommended for rich-media networks:

- **Class-based weighted fair queuing (CBWFQ):** CBWFQ involves a combination of bandwidth guarantee and dynamic fairness of other flows. It does not provide a latency guarantee and is suitable only for data traffic management. Figure 15-16 illustrates the CBWFQ process. In the event of congestion, the Layer 1 Tx ring for the interface fills up and pushes packets back into the Layer 3 CBWFQ queues (if configured). Each CBWFQ class is assigned its own queue. CBWFQ queues may also have a fair-queuing pre-sorter applied to fairly manage multiple flows contending for a single queue. In addition, each CBWFQ queue is serviced in a weighted round-robin (WRR) fashion, based on the bandwidth assigned to each class. The CBWFQ scheduler then forwards packets to the Tx ring.
- **Low-latency queuing (LLQ):** LLQ is a method that is essentially CBWFQ with strict priority. LLQ is suitable for mixes of data and real-time traffic. LLQ provides both latency and bandwidth guarantees. When LLQ is used in the CBWFQ system, it creates an extra priority queue in the WFQ system, which is serviced by a strict priority scheduler. Any class of traffic can therefore be attached to a service policy, which uses priority scheduling, and hence can be prioritized over other classes. In Figure 15-17, three real-time classes of traffic all funnel into the priority queue of LLQ, and other classes of traffic use the CBWFQ algorithm.

Figure 15-16 CBWFQ with Fair Queuing

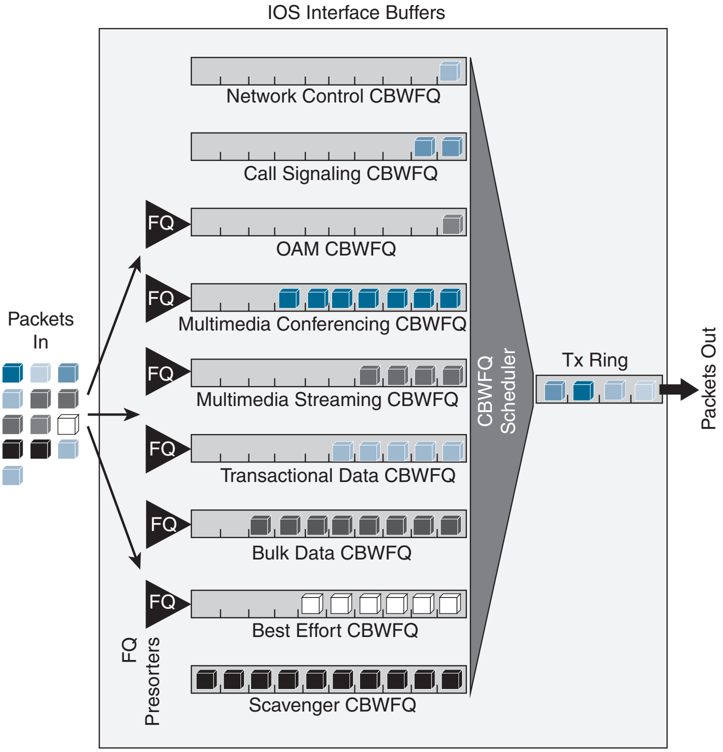
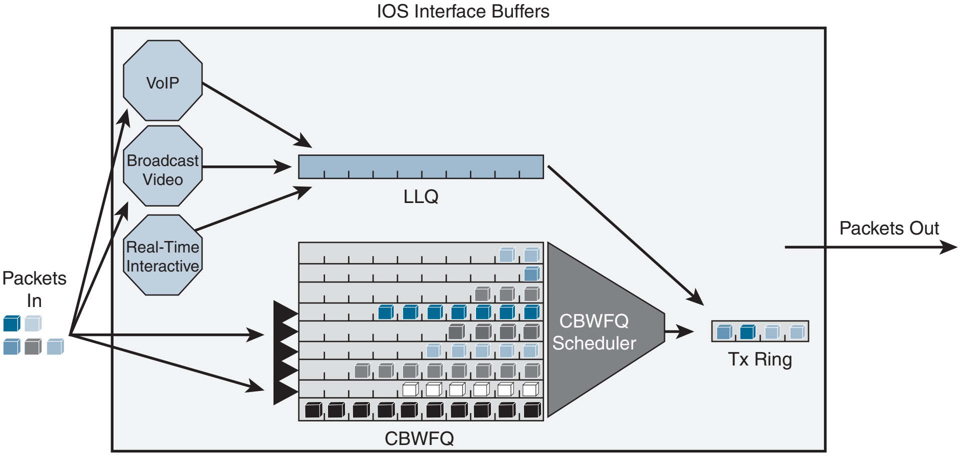

IOS Interface Buffers
Network Control CBWFQ
Call Signaling CBWFQ
FQ
OAM CBWFQ
FQ
Multimedia Conferencing CBWFQ
FQ
Multimedia Streaming CBWFQ
FQ
Transactional Data CBWFQ
FQ
Bulk Data CBWFQ
FQ
Best Effort CBWFQ
Scavenger CBWFQ
FQ Presorters
Packets In
CBWFQ Scheduler
Tx Ring
Packets Out

Figure 15-17 CBWFQ with LLQ


IOS Interface Buffers
VoIP
Broadcast Video
Real-Time Interactive
LLQ
Packets In
CBWFQ Scheduler
Tx Ring
Packets Out
CBWFQ

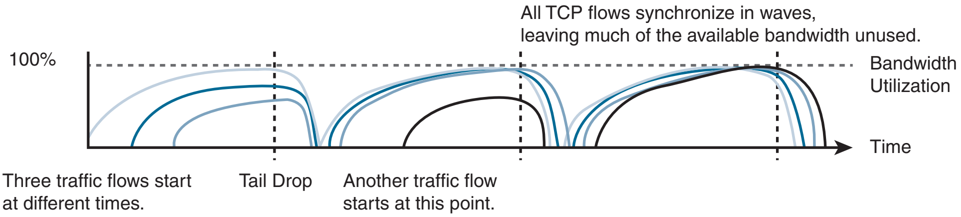
Tools for Congestion Avoidance

Queues are finite on any interface. Devices can either wait for queues to fill up and then start dropping packets or drop packets before the queues fill up. Dropping packets as they arrive is called *tail* *drop*. Selective dropping of packets while queues are filling up is called *congestion avoidance*. Queuing algorithms manage the front of a queue, and congestion mechanisms manage the back of the queue.

When a queue fills up, it drops packets as they arrive. This can result in waste of bandwidth if TCP traffic is predominant. Congestion avoidance drops random packets before a queue fills up. Randomly dropping packets instead of dropping them all at once, as it is done in a tail drop, prevents global synchronization of TCP streams. One mechanism that randomly drops packets is random early detection (RED). RED monitors the buffer depth and performs early discards (drops) on random packets when the minimum defined queue threshold is exceeded.

TCP has built-in flow control mechanisms that operate by increasing the transmission rates of traffic flows until packet loss occurs. When packet loss occurs, TCP drastically slows down the transmission rate and then again begins to increase the transmission rate. Because of TCP behavior, tail drop of traffic can result in suboptimal bandwidth utilization. TCP global synchronization is a phenomenon that can happen to TCP flows during periods of congestion because the senders reduce the transmission rate at the same time when packet loss occurs. TCP global synchronization is illustrated in Figure 15-18.

Figure 15-18 TCP Global Synchronization



Instead of supporting RED, Cisco IOS Software supports weighted random early detection (WRED). The principle is the same as with RED, except that the traffic weights skew the randomness of the packet drop. In other words, traffic that is more important is less likely to be dropped than less important traffic.

QoS Policy

QoS features can be applied using the Modular QoS command-line interface (CLI) (MQC). The MQC allows you to define a traffic class, create a traffic policy (policy map), and attach the traffic policy to an interface. The traffic policy contains the QoS feature that will be applied to the traffic class.

Define an Overall QoS Policy

The MQC structure allows you to define a traffic class, create a traffic policy, and attach the traffic policy to an interface.

Defining an overall QoS policy involves these three high-level steps:

1. Define a traffic class by using the **class-map** command. A traffic class is used to classify traffic.
2. Create a traffic policy by using the **policy-map** command. The terms *traffic policy* and *policy map* are often synonymous. A traffic policy (policy map) contains a traffic class and one or

more QoS features that are applied to the traffic class. The QoS features in the traffic policy determine how to treat the classified traffic.

3. Attach the traffic policy (policy map) to the interface by using the **service-policy** command.

Methods for Implementing a QoS Policy

In the past, the only way to configure individual QoS policies at each interface in a network was by using the command-line interface (CLI). Cutting and pasting configurations from one interface to another can ease administration. But this process is error-prone and time-consuming.

MQC

To simplify QoS configuration Cisco introduced the Modular QoS CLI (MQC). The MQC provides a single-module building-block approach to apply a policy to multiple interfaces. Example 15-2 shows a simple MQC policy configuration.

Example 15-2 Cisco MQC Example

```
Router
class-map match-any EMAIL
 match protocol exchange
 match protocol pop3
 match protocol smtp
 match protocol imap
class-map match-any WEB
 match protocol http
 match protocol secure-http
class-map match-all VOICE
 match protocol rtp audio
class-map match-all SCAVANGER
 match protocol netflix
!
policy-map MYMAP
 class EMAIL
  bandwidth 512
 class VOICE
  priority 256
 class WEB
  bandwidth 768
 class SCAVANGER
  police 128000
!
interface Serial0/1/0
service-policy output MYMAP
```

In this example, four class maps are configured: EMAIL, WEB, VOICE, and SCAVANGER. Each class map matches specific protocols that are identified using Cisco NBAR. A policy map named MYMAP is created to tie in each class map and define specific bandwidth requirements. For example, the EMAIL class map is guaranteed a minimum of 512 Kbps, and the WEB class map is guaranteed a minimum of 768 Kbps. Both of these will be processed using CBWFQ. The VOICE class map is configured using the **priority** keyword, which enables LLQ for voice traffic with a maximum of 256 Kbps. Finally, the SCAVANGER class is policed up to 128 Kbps. Traffic exceeding that speed is dropped. The MYMAP policy map is then applied outbound on Serial 0/1/0 to process packets leaving that interface.

Cisco AutoQoS

Instead of manually entering QoS policies at the CLI, an innovative technology known as Cisco AutoQoS simplifies the challenges of network administration by reducing QoS complexity, deployment time, and cost to enterprise networks. Cisco AutoQoS incorporates value-added intelligence in Cisco IOS Software and Cisco Catalyst software to provision and assist in the management of large-scale QoS deployments. Default Cisco validated QoS policies can be quickly implemented with Cisco AutoQoS.

Cisco DNA Center Application Policies

Today, you can configure QoS in an intent-based network by using application policies in Cisco DNA Center. Application policies comprise these basic parameters:

- **Application sets:** Application sets are sets of applications with similar network traffic needs. Each application set is assigned a business-relevant group (business relevant, default, or business irrelevant) that defines the priority of its traffic. QoS parameters in each of the three groups are defined based on Cisco Validated Design (CVD). You can modify some of these parameters to more closely align with your objectives. A business-relevant group classifies a given application set according to how relevant it is to the business and operations. The three business-relevant groups essentially map to three types of traffic: high priority, neutral, and low priority.

- **Site scope:** Site scope specifies the sites to which an application policy is applied. If you configure a wired policy, the policy is applied to all the wired devices in the site scope. Likewise, if you configure a wireless policy for a selected service set identifier (SSID), the policy is applied to all of the wireless devices with the SSID defined in the scope.

Cisco DNA Center translates these parameters into the proper device CLI commands. When you deploy the QoS policy, Cisco DNA Center configures these commands on the devices defined in the site scope. Cisco DNA Center configures QoS policies on devices based on the QoS feature sets available on the devices.

You can configure relationships between applications such that when traffic from one application is sent to another application (thus creating a specific a-to-b traffic flow), the traffic is handled in a specific way. The applications in this relationship are called *producers* and *consumers* and are defined as follows:

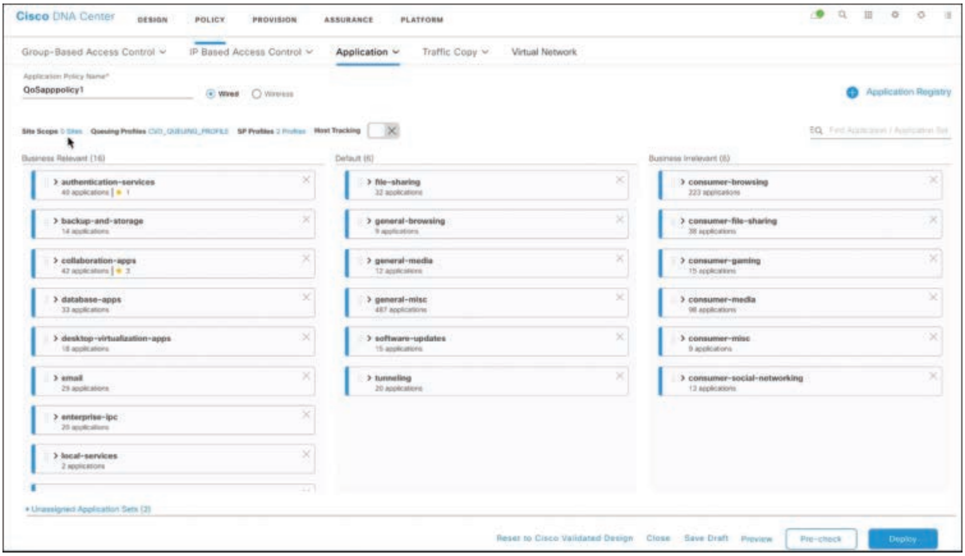
- **Producer:** This is the sender of the application traffic. For example, in a client/server architecture, the application server is considered the producer because the traffic primarily flows in the server-to-client direction. In the case of a peer-to-peer application, the remote peer is considered the producer.

- **Consumer:** This is the receiver of the application traffic. The consumer may be a client endpoint in a client/server architecture, or it may be the local device in a peer-to-peer application. Consumers may be endpoint devices, but they may, at times, be specific users of such devices (typically identified by IP addresses or specific subnets). There may also be times when an application is the consumer of another application's traffic flows.

Setting up a producer/consumer relationship allows you to configure specific service levels for traffic matching a particular scenario.

Figure 15-19 illustrates the Cisco DNA Center Policy Application dashboard. Notice the three business-relevant groups and the different QoS application policies under each column. You can easily modify these default settings by simply dragging and dropping a policy into the correct group.

Figure 15-19 Cisco DNA Center Policy Application Dashboard



Application Policy Name*: QoSapppolicy1

● Wired ○ Wireless

Cisco DNA Center
DESIGN
POLICY
PROVISION
ASSURANCE
PLATFORM
Group-Based Access Control
IP Based Access Control
Application
Traffic Copy
Virtual Network
Application Registry
Site Scope
Queuing Profiles
SP Profiles 2 Profiles
Host Tracking
Business Relevant (16)
authentication-services
40 applications | 1
backup-and-storage
14 applications
collaboration-apps
42 applications | 3
database-apps
33 applications
desktop-virtualization-apps
18 applications
email
29 applications
enterprise-ipc
20 applications
local-services
2 applications
Default (6)
file-sharing
32 applications
general-browsing
9 applications
general-media
12 applications
general-misc
487 applications
software-updates
15 applications
tunneling
20 applications
Business Irrelevant (6)
consumer-browsing
223 applications
consumer-file-sharing
38 applications
consumer-gaming
15 applications
consumer-media
98 applications
consumer-misc
9 applications
consumer-social-networking
13 applications
Unassigned Application Sets (2)
Reset to Cisco Validated Design
Close
Save Draft
Preview
Pre-check
Deploy

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	14
End-to-End QoS Network Design: Quality of Service for Rich-Media & Cloud Networks, second edition	3, 4, 5
Enterprise QoS Solution Reference Network Design Guide	https://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/WAN_and_MAN/QoS_SRND/QoS-SRND-Book/QoSIntro.html

Day 14

Network Assurance, Part 1

ENCOR 350-401 Exam Topics

Network Assurance

- Diagnose network problems using tools such as debugs, conditional debugs, trace route, ping, SNMP, and syslog
- Configure and verify SPAN/RSPAN/ERSPAN
- Configure and verify IPSLA

Key Topics

Today we start our review of concepts related to network assurance. Network outages that cause business-critical applications to become inaccessible could potentially cause an organization to sustain significant financial losses. Network engineers are often asked to perform troubleshooting in these cases. Troubleshooting is the process of responding to a problem that leads to its diagnosis and resolution.

Today we first examine the diagnostic principles of troubleshooting and how they fit into the overall troubleshooting process. We also explore various Cisco IOS network tools used in the diagnostic phase to assist in monitoring and troubleshooting an internetwork. We also look at how to use network analysis tools such as Cisco IOS CLI troubleshooting commands, as well as Cisco IP service-level agreements (SLAs) and different implementations of switched port analyzer services, such as Switched Port Analyzer (SPAN), Remote SPAN (RSPAN), and Encapsulated Remote SPAN (ERSPAN).

On Day 13, "Network Assurance, Part 2," we will discuss network logging services that can collect information and produce notifications of network events, such as syslog, Simple Network Management Protocol (SNMP), and Cisco NetFlow. These services are essential in maintaining network assurance and high availability of network services for users.

Troubleshooting Concepts

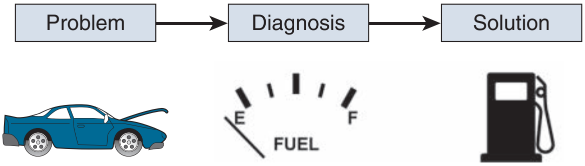
In general, the troubleshooting process starts when someone reports a problem. In a way, you could say that a problem does not exist until it is noticed, considered a problem, and reported. You need to differentiate between a problem, as experienced by a user, and the cause of that problem. So, the time that a problem was reported is not necessarily the same as the time at which the event that caused that problem occurred. However, a reporting user generally equates the problem with the symptoms, while the troubleshooter equates the problem with the root cause.

If the Internet connection flaps on a Saturday in a small company outside operating hours, is that a problem? Probably not, but it is very likely that it will turn into a problem on Monday morning if it is not fixed by then.

Although the distinction between symptoms and the cause may seem philosophical, it is good to be aware of the communication issues that can potentially arise.

A troubleshooting process starts with reporting and defining a problem, as illustrated in Figure 14-1. It is followed by the process of diagnosing the problem. During this process, information is gathered, the problem definition is refined, and possible causes for the problem are proposed. Eventually, this process should lead to a diagnosis of the root cause of the problem.

Figure 14-1 Basic Troubleshooting Steps



When the root cause has been found, possible solutions need to be proposed and evaluated. After the best solution is chosen, that solution should be implemented. Sometimes, the solution cannot be immediately implemented, and you need to propose a workaround until the solution can be implemented. The difference between a solution and a workaround is that a solution resolves the root cause of the problem, and a workaround only remedies or alleviates the symptoms of the problem.

Once the problem is fixed, all changes should be well documented. This information will be helpful the next time someone needs to resolve similar issues.

Diagnostic Principles

Although problem reporting and resolution are essential elements of the troubleshooting process, most troubleshooting time is spent in the diagnostic phase.

Diagnosis is the process in which you identify the nature and the cause of a problem. The essential elements of the diagnosis process are as follows:

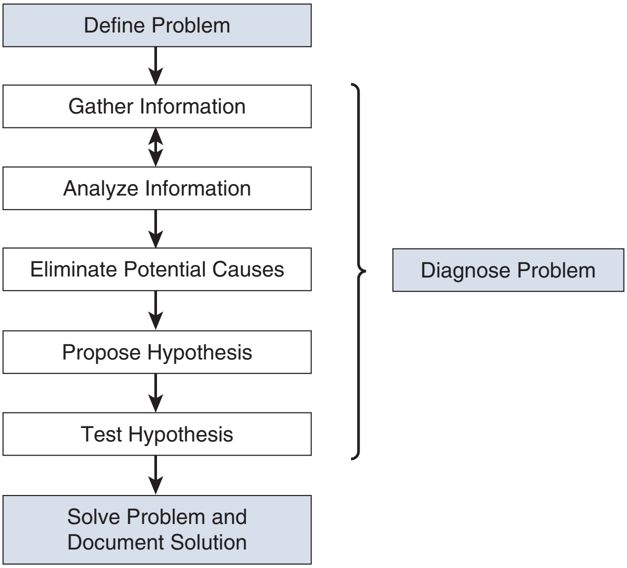
- **Gathered information:** Gathering information about what is happening is essential to the troubleshooting process. Usually, the problem report does not contain enough information for you to formulate a good hypothesis without first gathering more information. You can gather information and symptoms either directly by observing processes or indirectly by executing tests.

- **Analysis:** The gathered information is analyzed. Compare the symptoms against your knowledge of the system, processes, and baseline to separate the normal behavior from the abnormal behavior.
- **Elimination:** By comparing the observed behavior against expected behavior, you can eliminate possible problem causes.
- **Proposed hypotheses:** After gathering and analyzing information and eliminating the possible causes, you are left with one or more potential problem causes. You need to assess the probability of each of these causes so that you can propose the most likely cause as the hypothetical cause of the problem.
- **Testing:** Test the hypothetical cause to confirm or deny that it is the actual cause. The simplest way to perform testing is to propose a solution that is based on this hypothesis, implement that solution, and verify whether it solves the problem. If this method is impossible or disruptive, the hypothesis can be strengthened or invalidated by gathering and analyzing more information.

Network Troubleshooting Procedures: Overview

A troubleshooting method is a guiding principle that determines how you move through the phases of the troubleshooting process, as illustrated in Figure 14-2.

Figure 14-2 Troubleshooting Process


Define Problem
Gather Information
Analyze Information
Eliminate Potential Causes
Propose Hypothesis
Test Hypothesis
Solve Problem and Document Solution
Diagnose Problem

In a typical troubleshooting process for a complex problem, you continually move between the different processes: Gather some information, analyze it, eliminate some possibilities, gather more information, analyze again, formulate a hypothesis, test it, reject it, eliminate some more possibilities, gather more information, and so on. However, the time spent on each of these phases and the method of moving from phase to phase can be significantly different from person to person and is a key differentiator between effective and less-effective troubleshooters.

If you do not use a structured approach but move between the phases randomly, you might eventually find the solution, but the process will be very inefficient. In addition, if your approach has no structure, it is practically impossible to hand it over to someone else without losing all the progress that was made up to that point. You also may need to stop and restart your own troubleshooting process.

A structured approach to troubleshooting (no matter what the exact method) yields more predictable results in the end and makes it easier to pick up the process where you left off in a later stage or to hand it over to someone else.

Network Diagnostic Tools

This section focuses on the use of the **ping**, **traceroute**, and **debug** IOS commands.

Using the ping Command

Using the **ping** command is a very common method for troubleshooting the accessibility of devices. **ping** uses a series of Internet Control Message Protocol (ICMP) echo request and echo reply messages to determine the following:

- Whether a remote host is active or inactive
- The round-trip delay in communicating with the host
- Packet loss

The **ping** command first sends an echo request packet to an address and waits for a reply. The ping is successful only if the following two conditions are true:

- The echo request gets to the destination.
- The destination can get an echo reply to the source within a predetermined time called a timeout. The default value of this timeout is 2 seconds on Cisco routers.

Table 14-1 lists the possible responses when conducting a ping test.

Table 14-1 Ping Characters

Character	Description
!	Each exclamation point indicates receipt of a reply.
.	Each period indicates a network server timed out while waiting for a reply.
U	A destination unreachable error PDU was received.
Q	Source quench (destination too busy).
M	Could not fragment.
?	Unknown packet type.
&	Packet lifetime exceeded.

In Example 14-1, R1 has successfully tested its connectivity with a device at address 10.10.10.2.

Example 14-1 Testing Connectivity with *ping*

```
R1# ping 10.10.10.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.10.10.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/201/1003 ms
R1#
```

When using the **ping** command, it is possible to specify options to help in the troubleshooting process. Example 14-2 shows some of these options.

Example 14-2 Ping Options

```
R1# ping 10.10.10.2 ?
  Extended-data   specify extended data pattern
  data            specify data pattern
  df-bit          enable do not fragment bit in IP header
  repeat          specify repeat count
  size            specify datagram size
  source          specify source address or name
  timeout         specify timeout interval
  tos             specify type of service value
  validate        validate reply data
  <cr>
```

The most useful options are **repeat** and **source**. The **repeat** keyword allows you to change the number of pings sent to the destination instead of using the default value of five. The **source** keyword allows you to change the interface used as the source of the ping. By default, the source interface is the router's outgoing interface, based on the routing table. It is often desirable to test reachability from a different source interface instead.

The Extended Ping

The extended ping is used to perform a more advanced check of host reachability and network connectivity. To enter extended ping mode, type the **ping** keyword and immediately press Enter. Table 14-2 lists the options in an extended ping.

Table 14-2 Extended Ping Options

Field	Description
Protocol [ip]:	Prompts for a supported protocol (ip or ipv6). The default is ip.
Target IP address:	Prompts for the IP or IPv6 address or hostname of the destination node you plan to ping.
Repeat count [5]:	Number of ping packets sent to the destination address. The default is 5.

Field	Description
Datagram size [100]:	Size of the ping packet (in bytes). The default is 100 bytes.
Timeout in seconds [2]:	Timeout interval. The default is 2 (seconds). The ping is declared successful only if the echo reply packet is received before this time interval.
Extended commands [n]:	Specifies whether a series of additional commands appears. The default is no. Entering **y** displays the following extra options.
Source address or interface:	The interface or IP address of the router to use as a source address for the probes. The router normally picks the IP address of the outbound interface to use. The interface can also be mentioned, but it needs to have the correct syntax, as shown here: Source address or interface: Ethernet0/0.
Type of service [0]:	Specifies the Type of Service (ToS). The requested ToS is placed in each probe, but there is no guarantee that all routers will process the ToS. It is the Internet service's quality selection. The default is 0.
Set DF bit in IP header? [no]:	Specifies whether the Don't Fragment (DF) bit is to be set on the ping packet. If yes is specified, the Don't Fragment option does not allow this packet to be fragmented when it has to go through a segment with a smaller maximum transmission unit (MTU), and you receive an error message from the device that wanted to fragment the packet. This is useful for determining the smallest MTU in the path to a destination. The default is no.
Validate reply data? [no]:	Specifies whether to validate the reply data. The default is no.
Data pattern [0xABCD]	Specifies the data pattern. Different data patterns are used to troubleshoot framing errors and clocking problems on serial lines. The default is 0xABCD.
Loose, Strict, Record, Timestamp, Verbose[none]:	IP header options. This prompt offers more than one option that can be selected: **Verbose** is automatically selected along with any other option. **Record** is a very useful option because it displays the addresses of the hops (up to nine) the packet goes through. **Loose** allows you to influence the path by specifying the addresses of the hops you want the packet to go through. **Strict** is used to specify the hops that you want the packet to go through, but no other hops are allowed to be visited. **Timestamp** is used to measure round-trip time to particular hosts. The default is none.
Sweep range of sizes [n]:	Allows you to vary the sizes of the echo packets that are sent to determine the minimum sizes of the MTUs configured on the nodes along the path to the destination address. Performance problems caused by packet fragmentation are thus reduced. The default is no.
!!!!!	Each exclamation point (!) denotes receipt of a reply. A period (.) denotes that the network server timed out while waiting for a reply.
Success rate is 100 percent	Indicates the percentage of packets successfully echoed back to the router. Anything less than 80% is usually considered problematic.
round-trip min/avg/max = 1/2/4 ms	Indicates the round-trip travel time intervals for the protocol echo packets, including minimum/average/maximum (in milliseconds).

Example 14-3 shows R1 using the extended **ping** command to test connectivity with a device at address 10.10.50.2.

Example 14-3 Extended Ping Example

```
R1# ping
Protocol [ip]:
Target IP address: 10.10.50.2
Repeat count [5]: 1
Datagram size [100]:
Timeout in seconds [2]: 1
Extended commands [n]: y
Source address or interface:
Type of service [0]:
Set DF bit in IP header? [no]: y
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]: y
Sweep min size [36]: 1400
Sweep max size [18024]: 1500
Sweep interval [1]:
Type escape sequence to abort.
Sending 101, [1400..1500]-byte ICMP Echos to 10.10.50.2, timeout is 1 second:
Packet sent with the DF bit set
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!M.M.M.M.M.M.M.M.M.M.M.M.
Success rate is 76 percent (77/101), round-trip min/avg/max = 1/1/1 ms  <cr>
```

In this example, the extended ping is used to test the maximum MTU size supported across the network. The ping succeeds with a datagram size from 1400 to 1476 bytes and the Don't Fragment-bit (df-bit) set; for the rest of the sweep, the result is that the packets cannot be fragmented. This outcome can be determined because the sweep started at 1400 bytes, 100 packets were sent, and there was a 76% success rate (1400 + 76 = 1476).

For testing, you can sweep packets at different sizes (minimum, maximum), set the sweeping interval, and determine the MTU by seeing which packets are passing through the links and which packets need to be fragmented since you have already set the df-bit for all the packets.

Using traceroute

The **traceroute** tool is very useful if you want to determine the specific path that a packet takes to its destination. If there is an unreachable destination, you can determine where on the path the issue lies.

The **traceroute** command works by sending the remote host a sequence of three UDP datagrams with a TTL of 1 in the IP header and the destination ports 33434 (first packet), 33435 (second packet), and 33436 (third packet). The TTL of 1 causes the datagram to time out when it hits the first router in the path. The router responds with an ICMP "time exceeded" message, indicating that the datagram has expired.

The next three UDP datagrams are sent with TTL of 2 to destination ports 33437, 33438, and 33439.

After passing through the first router, which decrements the TTL to 1, the datagram arrives at the ingress interface of the second router. The second router drops the TTL to 0 and responds with an ICMP “time exceeded” message. This process continues until the packet reaches the destination and the ICMP “time exceeded” messages have been sent by all the routers along the path.

Since these datagrams are trying to access an invalid port at the destination host, ICMP “port unreachable” messages are returned when the packet reaches the destination, indicating an unreachable port; this event signals that the **traceroute** program is finished.

The possible responses when conducting a **traceroute** are displayed in Table 14-3.

Table 14-3 *traceroute* Characters

Character	Description
nn msec	For each node, the round-trip time, in milliseconds, for the specified number of probes
*	The probe timed out
A	Administratively prohibited (for example, by an access list)
Q	Source quench (destination too busy)
I	User interrupted test
U	Port unreachable
H	Host unreachable
N	Network unreachable
P	Protocol unreachable
T	Timeout
?	Unknown packet type

Example 14-4 shows R1 performing a **traceroute** to a device at address 10.10.20.1.

Example 14-4 Testing Connectivity with *traceroute*

```
R1# traceroute 10.10.20.1
Type escape sequence to abort.
Tracing the route to 10.10.20.1
VRF info: (vrf in name/id, vrf out name/id)
  1 10.10.50.1 1 msec 0 msec 1 msec
  2 10.10.40.1 0 msec 0 msec 1 msec
  3 10.10.30.1 1 msec 0 msec 1 msec
  4 10.10.20.1 1 msec *  2 msec
R1#
```

In this example, R1 is able to reach the device at address 10.10.20.1 in four hops. The first three hops represent Layer 3 devices between R1 and the destination, and the last hop is the destination itself.

As with **ping**, it is possible to add optional keywords to the **traceroute** command to influence its default behavior, and it is also possible to perform an extended **traceroute**, which operates in a similar way to the extended **ping** command.

Using Debug

The output from **debug** commands provides diagnostic information that includes various internetworking events relating to protocol status and network activity in general.

Use **debug** commands with caution. In general, it is recommended that these commands be used only when troubleshooting specific problems. Enabling debugging can disrupt the operation of the router when internetworks are experiencing high load conditions. Hence, if logging is enabled, the device may intermittently freeze when the console port gets overloaded with log messages.

Before you start a **debug** command, always consider the output that the **debug** command will generate and the amount of time it can take. Before debugging, you might want to look at your CPU load with the **show processes cpu** command. Verify that you have ample CPU available before you begin the debugging.

Cisco devices can display debugging output on various interfaces or can be configured to capture the debugging messages in a log:

- **Console:** By default, logging is enabled on the console port. Hence, the console port always processes debugging output, even if you are actually using some other port or method (such as AUX, vty, or buffer) to capture the output. Excessive debugging to the console port of a router can cause it to hang. You should consider changing where the debugging messages are captured and turn off logging to the console with the **no logging console** command. Some **debug** commands are very verbose, and, therefore, you cannot easily view any subsequent commands you wish to type while these commands are in process. To remedy the situation, configure **logging synchronous** on the console line.
- **AUX and vty ports:** To receive debugging messages when connected to the AUX port or when remotely logged in to the device via Telnet or SSH through the vty lines, type the command **terminal monitor**.
- **Logs:** Like syslog messages, debugging messages can be collected in logs. You can use the **logging** command to configure messages to be captured in an internal device buffer or an external syslog server.

The **debug ip packet** command helps you better understand the IP packet forwarding process, but this command only produces information on packets that are process switched by the router.

Packets that are forwarded through a router that is configured for fast switching or CEF are not sent to the processor, and hence the debugging does not display anything about those packets. To display packets forwarded through a router with the **debug ip packet** command, you need to disable fast switching on the router with the **no ip route-cache** command for unicast packets or **no ip mroute-cache** for multicast packets. These commands are configured on the interfaces where the traffic is supposed to flow. You can verify whether fast switching is enabled by using the **show ip interface** command.

The Conditional Debug

Another way of narrowing down the output of a **debug** command is to use a conditional **debug** command. If any debug condition commands are enabled, output is generated only for packets that contain information specified in the configured condition.

Table 14-4 lists the options available with a conditional **debug** command.

Table 14-4 Conditional Debug Options

Command	Description
called *dial-string*	Filters output on the basis of the called party number.
caller *dial-string*	Filters output on the basis of the calling party number.
calling *tidi/imsi-string*	Filters debug messages for General Packet Radio Service (GPRS) tunneling protocol (GTP) processing on the Gateway GPRS Support Node (GGSN) based on the tunnel identifier (TID) or international mobile system identifier (IMSI) in a Packet Data Protocol (PDP) Context Create Request message.
domain *domain-name*	Filters output on the basis of the specified domain.
interface *interface-id*	Filters output on the basis of the specified interface ID.
ip *ip-address*	Filters output on the basis of the specified IP address.
mac-address *hexadecimal-mac-address*	Filters messages on the specified MAC address.
portbundleip *ip-address*	Filters output on the basis of the port-bundle host key (PBHK) that uniquely identifies the session.
bundle *bundle-number*	Specifies the port bundle.
session-id *session-number*	Filters output on the specified Intelligent Service Architecture (ISA) session identifier.
username *username*	Filters output on the basis of the specified username.
vcid *vc-id*	Filters output on the basis of the specified VC ID.
vlan *vlan-id*	Filters output on the basis of the specified VLAN ID.
condition-id	ID number of the condition to be removed.
all	Removes all debugging conditions and conditions specified by the **debug condition interface** command. Use this keyword to disable conditional debugging and reenable debugging for all interfaces.

Example 14-5 shows the setting and verification of a debug condition for the GigabitEthernet 0/0/0 interface on R1. Any **debug** commands enabled on R1 would only produce logging output if there is a match on the GigabitEthernet 0/0/0 interface.

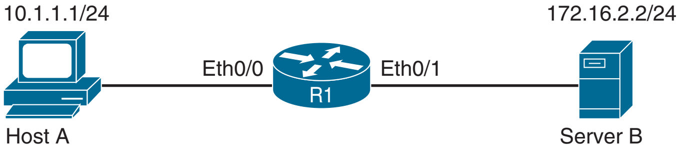
Example 14-5 Configuring and Verifying Conditional Debugs

```
R1# debug condition interface gigabitethernet 0/0/0
Condition 1 set
R1# show debug condition
Condition 1: interface Gi0/0/0 (1 flags triggered)
        Flags: Gi0/0/0
R1#
```

Another way of filtering debugging output is to combine the **debug** command with an access list. For example, with the **debug ip packet** command, you have the option to enter the name or number of an access list. Doing that causes the **debug** command to get focused only on packets that satisfy (that is, that are permitted by) the access list's statements.

In Figure 14-3, Host A uses Telnet to connect to Server B. You decide to use **debug** on the router connecting the segments where Host A and Server B reside.

Figure 14-3 Debugging with an Access List



Example 14-6 shows the commands used to test the Telnet session. Note that the **no ip route-cache** command was previously issued on R1's interfaces.

Example 14-6 Using the *debug* Command with an Access List

```
R1(config)# access-list 100 permit tcp host 10.1.1.1 host 172.16.2.2 eq telnet
R1(config)# access-list 100 permit tcp host 172.16.2.2 eq telnet host 10.1.1.1
   established
R1(config)# exit
R1# debug ip packet detail 100
IP packet debugging is on (detailed) for access list 100
```

```
HostA# telnet 172.16.2.2
Trying 172.16.2.2 ... Open

User Access Verification

Password:
ServerB>
```

```
R1
<. . . output omitted . . .>
*Jun  9 06:10:18.661: FIBipv4-packet-proc: route packet from Ethernet0/0 src
   10.1.1.1 dst 172.16.2.2
*Jun  9 06:10:18.661: FIBfwd-proc: packet routed by adj to Ethernet0/1 172.16.2.2
*Jun  9 06:10:18.661: FIBipv4-packet-proc: packet routing succeeded
*Jun  9 06:10:18.661: IP: s=10.1.1.1 (Ethernet0/0), d=172.16.2.2 (Ethernet0/1),
   g=172.16.2.2, len 43, forward
*Jun  9 06:10:18.661:      TCP src=62313, dst=23, seq=469827330, ack=3611027304,
   win=4064 ACK PSH
*Jun  9 06:10:18.661: IP: s=10.1.1.1 (Ethernet0/0), d=172.16.2.2 (Ethernet0/1),
   len 43, sending full packet
```

```
*Jun  9 06:10:18.661:      TCP src=62313, dst=23, seq=469827330, ack=3611027304,
  win=4064 ACK PSH
*Jun  9 06:10:18.662: IP: s=172.16.2.2 (Ethernet0/1), d=10.1.1.1, len 40, input
  feature
*Jun  9 06:10:18.662:      TCP src=23, dst=62313, seq=3611027304, ack=469827321,
  win=4110 ACK, MCI Check(108), rtype 0, forus FALSE, sendself FALSE, mtu 0,
  fwdchk FALSE
<. . . output omitted . . .>
```

Considering the addressing scheme used in Figure 14-3, access list 100 permits TCP traffic from Host A (10.1.1.1) to Server B (172.16.2.2) with the Telnet port (23) as the destination. Access list 100 also permits established TCP traffic from Server B to Host A. Using access list 100 with the **debug ip packet detail** command allows you to see only debugging packets that satisfy the access list. This is an effective troubleshooting technique that requires less overhead on your router while allowing all information on the subject you are troubleshooting to be displayed by the debugging facility.

Cisco IOS IP SLAs

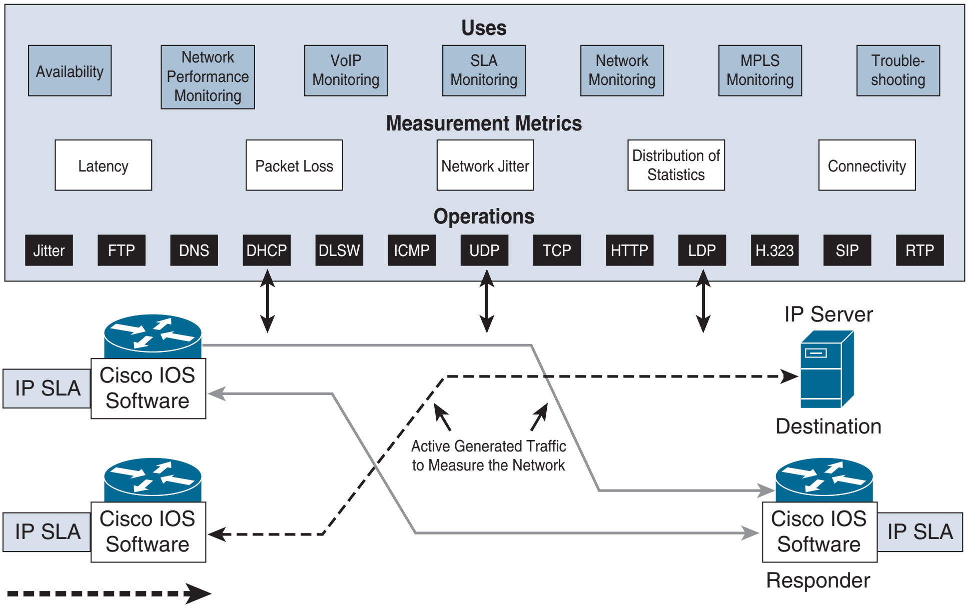
Network connectivity across the enterprise campus and also across the WAN and Internet from data centers to branch offices has become increasingly critical for customers, and any downtime or degradation can adversely affect revenue. Companies need some form of predictability with IP services. A service-level agreement (SLA) is a contract between a network provider and its customers or between a network department and its internal corporate customers. It provides a form of guarantee to customers about the level of user experience. An SLA typically outlines the minimum level of service and the expected level of service regarding network connectivity and performance for network users.

Typically, the technical components of an SLA contain a guaranteed level for network availability, network performance in terms of RTT, and network response in terms of latency, jitter, and packet loss. The specifics of an SLA vary depending on the applications that an organization is supporting in the network.

The tests generated by Cisco IOS devices used to determine whether an SLA is being met are called IP SLAs. IP SLA tests use various operations, as illustrated in Figure 14-4:

- FTP
- ICMP
- HTTP
- SIP
- Others

Figure 14-4 Cisco IOS IP SLA


Uses
Availability
Network Performance Monitoring
VoIP Monitoring
SLA Monitoring
Network Monitoring
MPLS Monitoring
Trouble-shooting
Measurement Metrics
Latency
Packet Loss
Network Jitter
Distribution of Statistics
Connectivity
Operations
Jitter
FTP
DNS
DHCP
DLSW
ICMP
UDP
TCP
HTTP
LDP
H.323
SIP
RTP
IP Server
IP SLA
Cisco IOS Software
Destination
Active Generated Traffic to Measure the Network
IP SLA
Cisco IOS Software
Cisco IOS Software
IP SLA
Responder

These IP SLA operations are used to gather many types of measurement metrics:

- Network latency and response time
- Packet loss statistics
- Network jitter and voice quality scoring
- End-to-end network connectivity

These measurement metrics provide network administrators with the information for various uses, including the following:

- Edge-to-edge network availability monitoring
- Network performance monitoring and network performance visibility
- VoIP, video, and VPN monitoring
- SLA monitoring
- IP service network health
- MPLS network monitoring
- Troubleshooting of network operations

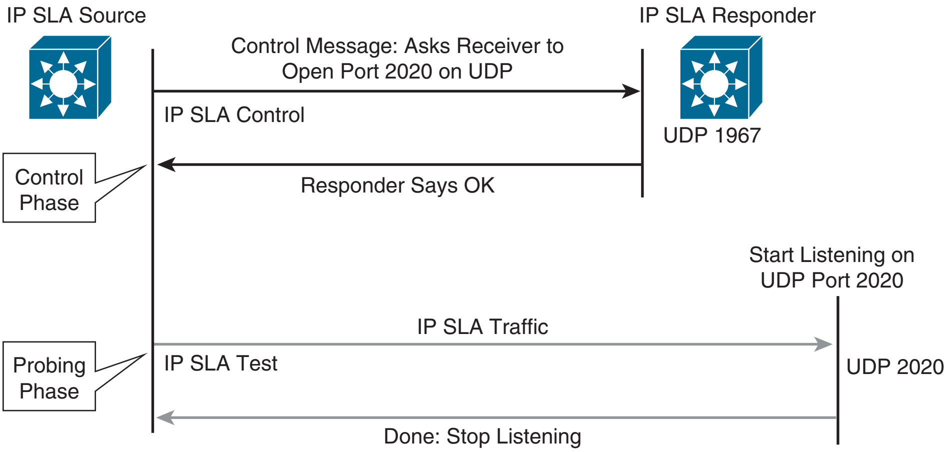
The networking department can use IP SLAs to verify that the service provider is meeting its own SLAs or to define service levels for its own critical business applications. An IP SLA can also be used as the basis for planning budgets and justifying network expenditures.

Administrators can ultimately reduce the mean time to repair (MTTR) by proactively isolating network issues. They can then change the network configuration, which is based on optimized performance metrics.

IP SLA Source and Responder

The IP SLA source is where all IP SLA measurement probe operations are configured either through the CLI or by using an SNMP tool that supports IP SLA operation. The IP SLA source is the Cisco IOS Software device that sends operational data, as shown in Figure 14-5.

Figure 14-5 Cisco IOS IP SLA Source and Responder


IP SLA Source
IP SLA Responder
Control Message: Asks Receiver to Open Port 2020 on UDP
IP SLA Control
UDP 1967
Control Phase
Responder Says OK
Start Listening on UDP Port 2020
IP SLA Traffic
IP SLA Test
UDP 2020
Probing Phase
Done: Stop Listening

The target device may or may not be a Cisco IOS Software device. Some operations require an IP SLA responder. The IP SLA source stores results in a Management Information Base (MIB). Reporting tools can then use SNMP to extract the data and report on it.

Tests performed on the IP SLA source are platform dependent, as shown in the following example:

```
Switch(config-ip-sla)# ?
IP SLAs entry configuration commands:
  dhcp          DHCP Operation
  dns           DNS Query Operation
  exit          Exit Operation Configuration
  ftp           FTP Operation
  http          HTTP Operation
  icmp-echo     ICMP Echo Operation
  path-echo     Path Discovered ICMP Echo Operation
  path-jitter   Path Discovered ICMP Jitter Operation
  tcp-connect   TCP Connect Operation
  udp-echo      UDP Echo Operation
  udp-jitter    UDP Jitter Operation
```

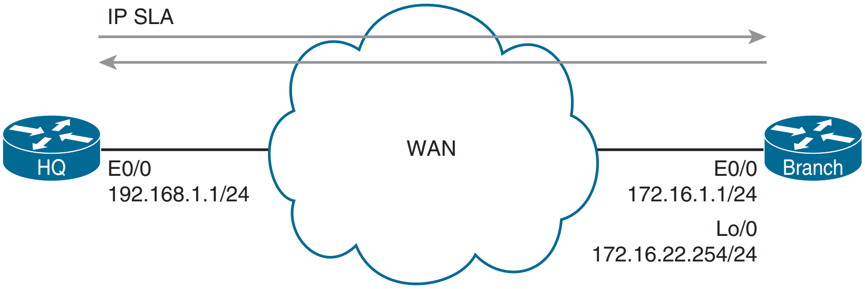
Although the destination of most of the tests can be any IP device, the measurement accuracy of some of the tests can be improved with an IP SLA responder.

An IP SLA responder is a Cisco IOS Software device that is configured to respond to IP SLA packets. An IP SLA responder adds a timestamp to the packets that are sent so that the IP SLA source can take into account any latency that occurs while the responder is processing the test packets. The response times that the IP SLA source records can, therefore, accurately represent true network delays.

It is important that the clocks on the source and on the responder be synchronized using NTP.

Figure 14-6 shows a simple topology to help illustrate the configuration process when deploying Cisco IOS IP SLA. In this example, two IP SLAs are configured: an ICMP echo SLA and a UDP jitter test. Both IP SLAs are sourced from the HQ router.

Figure 14-6 IP SLA Topology Example



Example 14-7 shows the commands used to configure both IP SLAs.

Example 14-7 Configuring Cisco IOS IP SLAs

```
HQ
ip sla 1
 icmp-echo 172.16.22.254
ip sla schedule 1 life forever start-time now
ip sla 2
 udp-jitter 172.16.22.254 65051 num-packets 20 interval 15
 request-data-size 160
 frequency 30
ip sla schedule 2 start-time now
```

```
Branch
ip sla responder
```

```
HQ# show ip sla summary
IPSLAs Latest Operation Summary
Codes: * active, ^ inactive, ~ pending
```

```
ID           Type        Destination       Stats        Return       Last
                                           (ms)         Code         Run
-----------------------------------------------------------------------
*1           icmp-echo   172.16.22.254     RTT=2        OK           50 seconds ago

*2           udp-jitter  172.16.22.254     RTT=1        OK           2 seconds ago


HQ# show ip sla statistics
IPSLAs Latest Operation Statistics

IPSLA operation id: 1
         Latest RTT: 3 milliseconds
Latest operation start time: 07:15:13 UTC Tue Jun 9 2020
Latest operation return code: OK
Number of successes: 10
Number of failures: 0
Operation time to live: Forever

IPSLA operation id: 2
Type of operation: udp-jitter
         Latest RTT: 1 milliseconds
Latest operation start time: 07:15:31 UTC Tue Jun 9 2020
Latest operation return code: OK
RTT Values:
         Number Of RTT: 20                RTT Min/Avg/Max: 1/1/4 milliseconds
Latency one-way time:
         Number of Latency one-way Samples: 19
         Source to Destination Latency one way Min/Avg/Max: 0/1/3 milliseconds
         Destination to Source Latency one way Min/Avg/Max: 0/0/1 milliseconds
Jitter Time:
         Number of SD Jitter Samples: 19
         Number of DS Jitter Samples: 19
         Source to Destination Jitter Min/Avg/Max: 0/1/3 milliseconds
         Destination to Source Jitter Min/Avg/Max: 0/1/1 milliseconds
<. . . output omitted . . .>
```

In the example, HQ is configured with two SLAs using the **ip sla** *operation-number* command. SLA number 1 is configured to send ICMP echo-request messages to the Loopback 0 IP address of the Branch router. IP SLA number 2 is configured for the same destination but has extra parameters: The destination UDP port is set to 65051, and every 30 seconds HQ will transmit 20 160-byte packets, which will be sent 15 milliseconds apart.

Both SLAs are then activated using the **ip sla schedule** command. The **ip sla schedule** command schedules when the test starts, how long it runs, and how long the collected data is kept. The syntax is as follows:

```
Router(config)# ip sla schedule operation-number [life {forever | seconds}] [start-
  time {hh:MM[:ss] [month day | day month] | pending | now | after hh:mm:ss}]
  [ageout seconds] [recurring]
```

With the **life** keyword, you set how long the IP SLA test runs. If you choose **forever**, the test runs until you manually remove it. By default, the IP SLA test runs for 1 hour.

With the **start-time** keyword, you set when the IP SLA test should start. You can start the test right away by issuing the **now** keyword, or you can configure a delayed start.

With the **ageout** keyword, you can control how long the collected data is kept.

With the **recurring** keyword, you can schedule a test to run periodically—for example, at the same time each day.

The Branch router is configured as an IP SLA responder. This is not required for SLA number 1, but it is required for SLA number 2.

You can use the **show ip sla summary** and **show ip sla statistics** commands to investigate the results of the tests. In this case, both SLAs are reporting an Ok status, and the UDP jitter SLA is gathering latency and jitter times between the HQ and Branch routers.

The IP SLA UDP jitter operation was designed primarily to diagnose network suitability for real-time traffic applications such as VoIP, video over IP, and real-time conferencing.

Jitter defines inter-packet delay variance. When multiple packets are sent consecutively from the source to the destination 10 milliseconds apart (for example), and the network is behaving ideally, the destination should receive the packets 10 milliseconds apart. But if there are delays in the network (such as queuing, arriving through alternate routes, and so on), the arrival delay between packets might be greater than or less than 10 milliseconds.

Switched Port Analyzer Overview

A traffic sniffer can be a valuable tool for monitoring and troubleshooting a network. Properly placing a traffic sniffer to capture a traffic flow but not interrupt it can be challenging.

When LANs were based on hubs, connecting a traffic sniffer was simple. When a hub receives a packet on one port, the hub sends out a copy of that packet on all ports except the one where the hub received the packet. A traffic sniffer that connected a hub port could thus receive all traffic in the network.

Modern local networks are essentially switched networks. After a switch boots, it starts to build up a Layer 2 forwarding table that is based on the source MAC address of the different packets that the switch receives. After this forwarding table is built, the switch forwards traffic destined for a MAC address directly to the corresponding port, thus preventing a traffic sniffer that is connected to another port from receiving the unicast traffic. The SPAN feature was therefore introduced on switches.

SPAN involves two different port types. The source port is a port that is monitored for traffic analysis. SPAN can copy ingress, egress, or both types of traffic from a source port. Both Layer 2 and Layer 3 ports can be configured as SPAN source ports. The traffic is copied to the destination (also called monitor) port.

The association of source ports and a destination port is called a *SPAN session*. In a single session, you can monitor at least one source port. Depending on the switch series, you might be able to copy session traffic to more than one destination port.

Alternatively, you can specify a source VLAN, where all ports in the source VLAN become sources of SPAN traffic. Each SPAN session can have either ports or VLANs as sources, but not both.

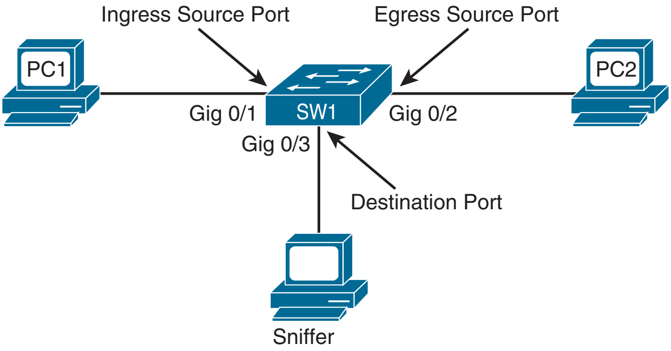
Local SPAN

A local SPAN session is an association of a source ports and source VLANs with one or more destination ports. You can configure local SPAN on a single switch. Local SPAN does not have separate source and destination sessions.

The SPAN feature allows you to instruct a switch to send copies of packets that are seen on one port to another port on the same switch.

To analyze the traffic flowing from PC1 to PC2, you need to specify a source port, as illustrated in Figure 14-7. You can either configure the GigabitEthernet 0/1 interface to capture the ingress traffic or the GigabitEthernet 0/2 interface to capture the egress traffic. Then you specify the GigabitEthernet 0/3 interface as a destination port. Traffic that flows from PC1 to PC2 is then copied to that interface, and you can analyze it with a traffic sniffer such as Wireshark or SolarWinds.

Figure 14-7 Local SPAN Example



Besides monitoring traffic on ports, you can also monitor traffic on VLANs.

Local SPAN Configuration

To configure local SPAN, associate the SPAN session number with source ports or VLANs and associate the SPAN session number with the destination, as shown in the following configuration:

```
SW1(config)# monitor session 1 source interface GigabitEthernet0/1
SW1(config)# monitor session 1 destination interface GigabitEthernet0/3
```

This example configures the GigabitEthernet 0/1 interface as the source and the GigabitEthernet 0/3 interface as the destination of SPAN session 1.

When you configure the SPAN feature, you must know the following:

- The destination port cannot be a source port or vice versa.
- The number of destination ports is platform dependent; some platforms allow for more than one destination port.
- The destination port is no longer a normal switch port; only monitored traffic passes through that port.

In the previous example, the objective is to capture all the traffic that is sent or received by the PC connected to the GigabitEthernet 0/1 port on the switch. A packet sniffer is connected to the GigabitEthernet 0/3 port. The switch is instructed to copy all the traffic that it sends and receives on GigabitEthernet 0/1 to GigabitEthernet 0/3 by configuring a SPAN session.

If you do not specify a traffic direction, the source interface sends both transmitted (Tx) and received (Rx) traffic to the destination port to be monitored. You have the ability to specify the following options:

- **Rx:** Monitor received traffic.
- **Tx:** Monitor transmitted traffic.
- **Both:** Monitor both received and transmitted traffic (default).

Verify the Local SPAN Configuration

You can verify the configuration of a SPAN session by using the **show monitor** command, as illustrated in Example 14-8:

Example 14-8 Verifying Local SPAN

```
SW1# show monitor
Session 1
------------
Type                      : Local Session
Source ports              :
  Both                    : Gi0/1
Destination ports         : Gi0/3
  Encapsulation           : Native
      Ingress   :  Disabled
```

As shown in this example, the **show monitor** command returns the type of the session, the source ports for each traffic direction, and the destination port. In the example, information about session number 1 is presented: The source port for both traffic directions is GigabitEthernet 0/1, and the destination port is GigabitEthernet 0/3. The ingress SPAN is disabled on the destination port, so only traffic that leaves the switch is copied to it.

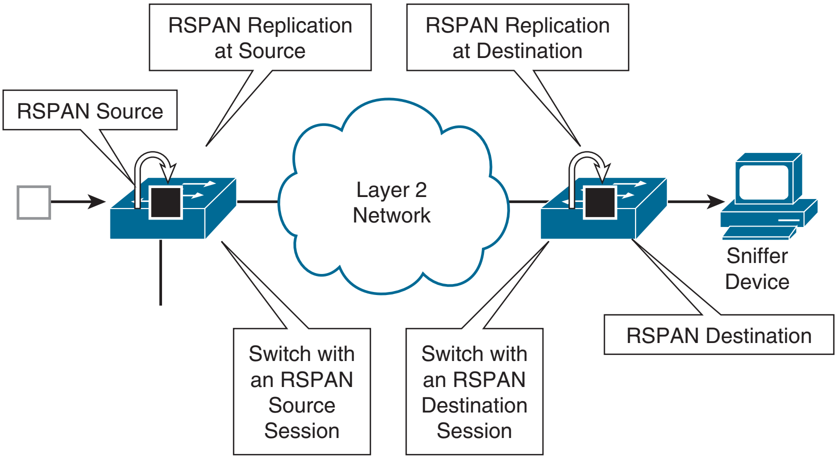
If you have more than one session configured, information about all sessions is shown after you use the **show monitor** command.

Remote SPAN

The local SPAN feature is limited because it allows for only a local copy on a single switch. A typical switched network usually consists of multiple switches, and it is possible to monitor ports spread all over the switched network with a single packet sniffer. This setup is possible with Remote SPAN (RSPAN).

Remote SPAN supports source and destination ports on different switches, and local SPAN supports only source and destination ports on the same switch. RSPAN consists of the RSPAN source session, RSPAN VLAN, and RSPAN destination session, as illustrated in Figure 14-8.

Figure 14-8 RSPAN


RSPAN Replication at Source
RSPAN Replication at Destination
RSPAN Source
Layer 2 Network
Sniffer Device
RSPAN Destination
Switch with an RSPAN Source Session
Switch with an RSPAN Destination Session

You separately configure the RSPAN source sessions and destination sessions on different switches. Your monitored traffic is flooded into an RSPAN VLAN that is dedicated for the RSPAN session in all participating switches. The RSPAN destination port can then be anywhere in that VLAN.

On some of the platforms, a reflector port needs to be specified together with an RSPAN VLAN. The reflector port is a physical interface that acts as a loopback and reflects the traffic that is copied from source ports to an RSPAN VLAN. No traffic is actually sent out of the interface that is assigned as the reflector port. The need for a reflector port is related to a hardware design limitation on some platforms. The reflector port can be used for only one session at a time. RSPAN supports source ports, source VLANs, and destinations on different switches, which provide remote monitoring of multiple switches across a network. RSPAN uses a Layer 2 VLAN to carry SPAN traffic between switches, which means there needs to be Layer 2 connectivity between the source and destination switches.

RSPAN Configuration

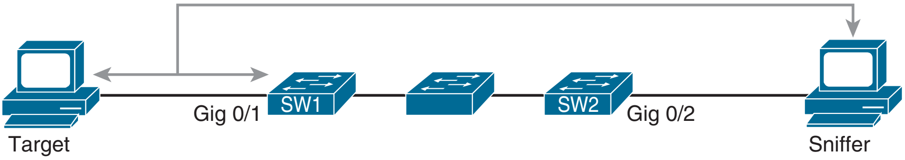
There are some differences between the configuration of RSPAN and the configuration of local SPAN. Example 14-9 shows the configuration of RSPAN. VLAN 100 is configured as the SPAN VLAN on SW1 and SW2. For SW1, the interface GigabitEthernet 0/1 is the source port in session 2, and VLAN 100 is the destination in session 2. For SW2, the interface GigabitEthernet 0/2 is the destination port in session 3, and VLAN 100 is the source in session 3. Session numbers are local to each switch, so they do not need to be the same on every switch.

Example 14-9 Configuring RSPAN

```
SW1(config)# vlan 100
SW1(config-vlan)# name SPAN-VLAN
SW1(config-vlan)# remote-span
SW1(config)# monitor session 2 source interface GigabitEthernet0/1
SW1(config)# monitor session 2 destination remote vlan 100

SW2(config)# vlan 100
SW2(config-vlan)# name SPAN-VLAN
SW2(config-vlan)# remote-span
SW2(config)# monitor session 3 destination interface GigabitEthernet0/2
SW2(config)# monitor session 3 source remote vlan 100
```

Figure 14-9 illustrates the topology for this example.

Figure 14-9 RSPAN Topology Example



Because the ports are now on two different switches, you use a special RSPAN VLAN to transport the traffic from one switch to the other. You configure this VLAN as you would any other VLAN, but you also enter the **remote-span** keyword in VLAN configuration mode. You need to define this VLAN on all switches in the path.

Verify the Remote SPAN Configuration

As with the local SPAN configuration, you can verify the RSPAN session configuration by using the **show monitor** command.

The only difference is that on the source switch, the session type is now identified as "Remote Source Session," while on the destination switch the type is marked as "Remote Destination Session," as shown in Example 14-10:

Example 14-10 Verifying Remote SPAN

```
SW1# show monitor
Session 2
------------
Type                     : Remote Source Session
Source ports             :
  Both                   : Gi0/1
Dest RSPAN VLAN          : 100
```

```
SW2# show monitor
Session 3
------------
Type                     : Remote Destination Session
Source RSPAN VLAN        : 100
Destination ports        : Gi0/2
  Encapsulation          : Native
      Ingress  :  Disabled
```

Encapsulated Remote SPAN

The Cisco-proprietary Encapsulated Remote SPAN (ERSPAN) mirrors traffic on one or more "source" ports and delivers the mirrored traffic to one or more "destination" ports on another switch. The traffic is encapsulated in Generic Routing Encapsulation (GRE) and is, therefore, routable across a Layer 3 network between the "source" switch and the "destination" switch. ERSPAN supports source ports, source VLANs, and destination ports on different switches, which provide remote monitoring of multiple switches across your network.

ERSPAN consists of an ERSPAN source session, routable ERSPAN GRE encapsulated traffic, and an ERSPAN destination session.

A device that has only an ERSPAN source session configured is called an ERSPAN source device, and a device that has only an ERSPAN destination session configured is called an ERSPAN termination device.

You separately configure ERSPAN source sessions and destination sessions on different switches.

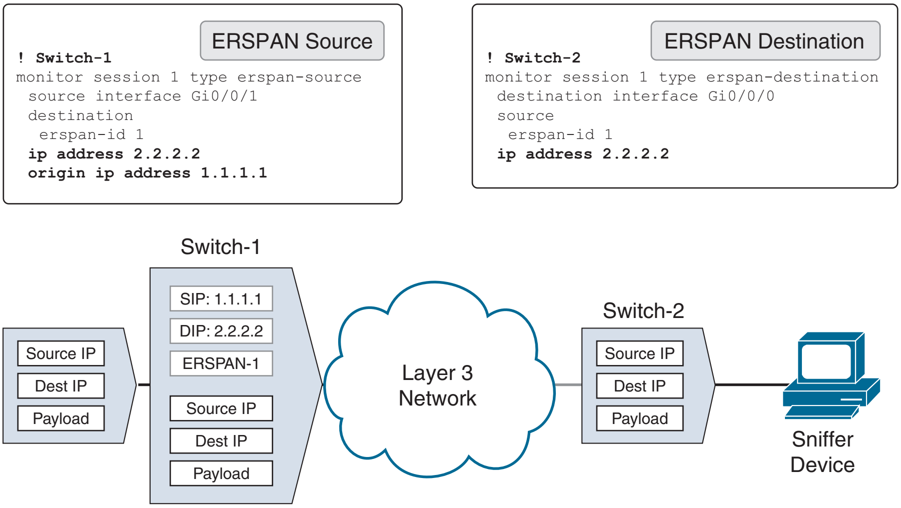
To configure an ERSPAN source session on one switch, you associate a set of source ports or VLANs with a destination IP address, ERSPAN ID number, and, optionally, a VRF name. To configure an ERSPAN destination session on another switch, you associate the destinations with the source IP address, ERSPAN ID number, and, optionally, a virtual routing and forwarding (VRF) name.

ERSPAN source sessions do not copy locally sourced RSPAN VLAN traffic from source trunk ports that carry RSPAN VLANs. ERSPAN source sessions do not copy locally sourced ERSPAN GRE-encapsulated traffic from source ports. Each ERSPAN source session can have either ports or VLANs as sources but not both. The ERSPAN source session copies traffic from the source ports or source VLANs and forwards the traffic using routable GRE-encapsulated packets to the ERSPAN destination session. The ERSPAN destination session switches the traffic to the destinations.

ERSPAN Configuration

The diagram in Figure 14-10 shows the configuration of ERSPAN session 1 between Switch-1 and Switch-2.

Figure 14-10 ERSPAN Configuration Example



```
! Switch-1
monitor session 1 type erspan-source
 source interface Gi0/0/1
 destination
  erspan-id 1
 ip address 2.2.2.2
 origin ip address 1.1.1.1
```

```
! Switch-2
monitor session 1 type erspan-destination
 destination interface Gi0/0/0
 source
  erspan-id 1
 ip address 2.2.2.2
```

On Switch-1, the **source interface** command associates the ERSPAN source session number with the source ports or VLANs and selects the traffic direction to be monitored. The **destination** command enters the ERSPAN source session destination configuration mode. **erspan-id** configures the ID number used by the source and destination sessions to identify the ERSPAN traffic, which must also be entered in the ERSPAN destination session configuration. The **ip address** command configures the ERSPAN flow destination IP address, which must also be configured on an interface on the destination switch and must be entered in the ERSPAN destination session configuration. The **origin ip address** command configures the IP address used as the source of the ERSPAN traffic.

On Switch-2, the **destination interface** command associates the ERSPAN destination session number with the destinations. The **source** command enters ERSPAN destination session source configuration mode. **erspan-id** configures the ID number used by the destination and destination sessions to identify the ERSPAN traffic. This must match the ID entered in the ERSPAN source session. The **ip address** command configures the ERSPAN flow destination IP address. This must be an address on a local interface and must match the address entered in the ERSPAN source session.

ERSPAN Verification

You can use the **show monitor** session command to verify the configuration, as shown in Example 14-11:

Example 14-11 Verifying ERSPAN

```
Switch-1# show monitor session 1
Session 1
---------
Type                    : ERSPAN Source Session
Status                  : Admin Enabled
Source Ports            :
    RX Only             : Gi0/0/1
Destination IP Address  : 2.2.2.2
MTU                     : 1464
Destination ERSPAN ID   : 1
Origin IP Address       : 1.1.1.1
```

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	**Module, Chapter, or Link**
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	24
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	11

Day 13

Network Assurance, Part 2

ENCOR 350-401 Exam Topics

Network Assurance

- Diagnose network problems using tools such as debugs, conditional debugs, trace route, ping, SNMP, and syslog
- Configure and verify device monitoring using syslog for remote logging
- Configure and verify NetFlow and Flexible NetFlow

Key Topics

Today we continue our review of concepts related to network assurance. We discuss network logging services that can collect information and produce notification of network events, such as syslog, Simple Network Management Protocol (SNMP), and Cisco NetFlow. These services are essential in maintaining network assurance and high availability of network services for users.

Logging Services

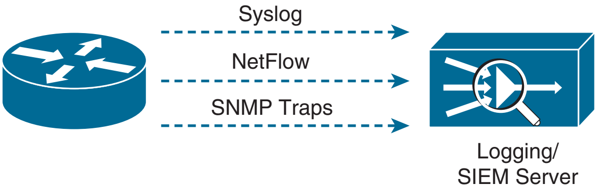
Network administrators need to implement logging to understand what is happening in the network—to detect unusual network traffic and network device failures and also to monitor what type of traffic traverses the network.

Logging can be implemented locally on a router, but this method is not scalable. In addition, if a router reloads, all the logs that are stored on the router are lost. Therefore, it is important to implement logging to an external destination, which can occur using various mechanisms, as illustrated in Figure 13-1:

- Cisco device syslog messages, which include OS notifications about unusual network activity or administrator-implemented debug messages
- SNMP trap notifications about network device status or configured thresholds being reached
- Exporting of network traffic flows using NetFlow

When implementing logging, it is important that dates and times be accurate and synchronized across all the network infrastructure devices. Without time synchronization, it is very difficult to correlate different sources of logging. NTP is typically used to ensure time synchronization across an enterprise network. (NTP is discussed on Day 21, "Network Services.")

Figure 13-1 Logging Services



Understanding Syslog

Network devices generate messages about different events. These messages are sent to an operating system process, which helps them proceed to the destination. Syslog is a protocol that allows a machine to send event notification messages across IP networks to event message collectors.

By default, a network device sends the output from system messages and debug-privileged EXEC commands to a logging process. The logging process controls the distribution of logging messages to various destinations. Syslog services provide a means to gather logging information for monitoring and troubleshooting, to select the type of logging information that is captured, and to specify the destinations of captured syslog messages.

Cisco devices can display syslog messages on various interfaces or can be configured to capture them in a log. The following are some of the options:

- **Console:** By default, logging is enabled on the console port. Hence, the console port always processes syslog output, even if you are actually using some other port or method (such as aux, vty, or buffer) to capture the output.

- **AUX and VTY ports:** To receive syslog messages when connected to the AUX port or remotely logged in to a device via Telnet or SSH through the vty lines, enter the **terminal monitor** command.

- **Memory buffer:** Memory log messages can be logged to an internal buffer. When the buffer is filled, newer messages overwrite older messages. The buffer size can be changed, but to prevent a router from running out of memory, it is important to avoid making the buffer size too large. To enable system message logging to a local buffer, use the **logging buffered** command in global configuration mode. To display messages that are logged in the buffer, use the **show logging** command. The first message displayed is the oldest message in the buffer.

- **Syslog server:** To log system messages and debug output to a remote host, use the **logging host** *ip-address* command in global configuration mode. This command identifies the IP address of a remote host (usually a device serving as a syslog server) to receive logging messages. By issuing this command more than once, you can build a list of hosts that receive logging messages.

- **Flash memory:** Logging to a buffer can be problematic when you are trying to capture debugs for an intermittent issue or when traffic is heavy because when the buffer is full, older messages are overwritten, and when the device reboots, all messages are lost. Using persistent logging allows you to write logged messages to files on a router's flash disk. To log messages to flash, use the **logging persistent** command.

Syslog Message Format and Severity

By default, Cisco IOS Software generates syslog messages with the following format:

```
seq no:time stamp: %facility-severity-MNEMONIC:description
```

Table 13-1 describes the elements of the Cisco IOS Software syslog message.

Table 13-1 Syslog Message Format

Element	Description
seq no	Stamps log messages with a sequence number only if the **service sequence-numbers** global configuration command is used.
time stamp	Indicates the date and time of the message or event, which appears only if the **service time stamps log** [*datetime* | *log*] global configuration command is used.
facility	Indicates the facility to which the message refers (for example, Simple Network Management Protocol [SNMP], system, and so on).
severity	Specifies the severity of the message, from 0 (highest) to 7 (lowest).
MNEMONIC	Uniquely describes the message in a text string.
description	Provides detailed information about the event that the message is reporting.

The following is an example of a syslog message that is informing the administrator that FastEthernet0/22 came up:

```
*Apr 22 11:05:55.423: %LINEPROTO-5-UPDOWN: Line protocol on Interface
  FastEthernet0/22, changed state to up
```

There are eight levels of severity for logging messages, numbered 0 to 7, from most severe (emergency messages) to least severe (debug messages). Table 13-2 lists the levels. By default, system logging is on, and the default severity level is debugging (severity 7), which means that all messages are logged.

Table 13-2 Syslog Severity Levels

Severity Level	Description
Emergency (severity 0)	System is unusable
Alert (severity 1)	Immediate action needed
Critical (severity 2)	Critical condition
Error (severity 3)	Error condition

Severity Level	Description
Warning (severity 4)	Warning condition
Notification (severity 5)	Normal but significant condition
Informational (severity 6)	Informational message
Debugging (severity 7)	Debugging message

To limit messages logged based on severity, use the **logging trap** *level* command in global configuration mode. If severity level 0 is configured, only emergency messages will be displayed. If, on the other hand, severity level 4 is configured, all messages with severity levels up to 4 (emergency, alert, critical, error, and warning) will be displayed.

A lot of information can be displayed at the lowest severity level (7, for debugging messages), which can even hamper the performance of your network. Use this level with caution.

Simple Network Management Protocol

Simple Network Management Protocol (SNMP) has become the standard for network management. It is a simple, easy-to-implement protocol and is supported by nearly all vendors. SNMP defines how management information is exchanged between SNMP managers and SNMP agents. It uses UDP as a transport mechanism to retrieve and send management information, such as Management Information Base (MIB) variables.

SNMP is typically used to gather environment and performance data, such as device CPU usage, memory usage, interface traffic, interface error rate, and so on.

SNMP has two main components:

- **SNMP manager, also known as a network management station (NMS):** The manager collects management data from managed devices via polling or trap messages.
- **SNMP agent:** The agent runs on a managed network device, locally organizing data and sending it to NMS.

The SNMP manager periodically polls the SNMP agents on managed devices for data. Periodic polling has a disadvantage: There is a delay between the occurrence of an event and when the SNMP manager polls the data.

SNMP agents on managed devices collect device information and translate it into a compatible SNMP format, according to the MIB. An MIB is a collection of definitions of managed objects. SNMP agents keep a database of values for definitions written in the MIB.

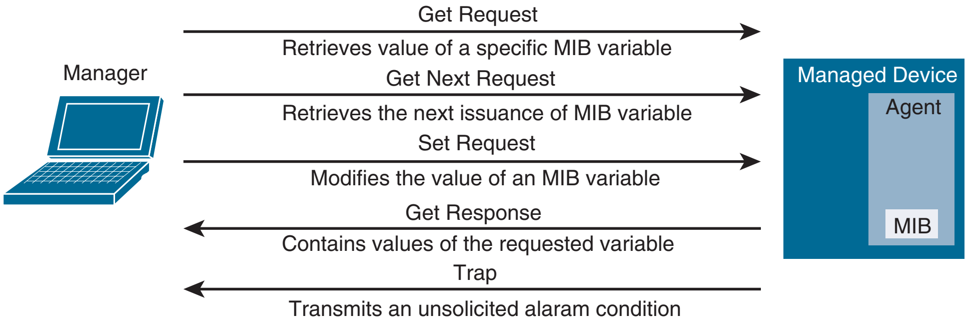
Agents also generate SNMP traps, which are unsolicited notifications that are sent from agent to manager. SNMP traps are event based and provide near-real-time event notifications. The idea behind trap-directed notification is that if an SNMP manager is responsible for a large number of devices, and each device has a large number of SNMP objects that are being tracked, it is impractical for the SNMP manager to poll or request information from every SNMP object on every device. The solution is for each SNMP agent on the managed device to notify the manager without solicitation. An agent does this by sending a message known as a *trap*. Trap-directed notification can result in substantial savings of network and agent resources by eliminating the need for frivolous SNMP requests. However, it is not possible to totally eliminate SNMP polling. SNMP requests are required for discovery and topology changes. In addition, a managed device agent cannot send a trap if the device has had a catastrophic outage.

Free and enterprise network management server software bundles provide data collection, storage, manipulation, and presentation services. A network management server offers a look into historical data and anticipated trends. An NMS triggers alarms based on SNMP values. The central NMS view provides an overview of the entire network so a network operator can easily identify irregular events, such as increased traffic and device unavailability due to DoS attacks.

SNMP Operations

SNMPv1 introduced five message types: Get Request, Get Next Request, Set Request, Get Response, and Trap (see Figure 13-2). New functionality has been added to subsequent versions of SNMP.

Figure 13-2 SNMP Message Types


Get Request
Retrieves value of a specific MIB variable
Manager
Get Next Request
Retrieves the next issuance of MIB variable
Set Request
Modifies the value of an MIB variable
Get Response
Contains values of the requested variable
Trap
Transmits an unsolicited alaram condition
Managed Device
Agent
MIB

SNMPv2 introduced two new message types: Get Bulk Request, which polls large amounts of data, and Inform Request, a type of trap message with expected acknowledgment on receipt. SNMPv2 also added 64-bit counters to accommodate faster network interfaces.

SNMPv2 also added a complex security model, which was never widely accepted. Instead, a "lighter" version of SNMPv2, known as Version 2c, was introduced. Due to its wide acceptance, SNMPv2c is now considered the de facto Version 2 standard.

SNMPv3 added methods to ensure the secure transmission of critical data between the manager and agent. It provides flexibility in defining security policy. You can define a secure policy per group, and you can optionally limit the IP addresses to which a group's members can belong. You have to define encryption and hashing algorithms and passwords for each user.

SNMPv3 introduced three levels of security:

- **noAuthNoPriv:** No authentication is required, and no privacy (encryption) is provided.
- **authNoPriv:** Authentication is based on MD5 or SHA. No encryption is provided.
- **authPriv:** In addition to authentication, CBC-DES encryption is used.

There are some basic guidelines you should follow when setting up SNMP in a network:

- **Restrict access to read-only:** NMS systems rarely need SNMP write access. Separate community credentials should be configured for systems that require write access.
- **Restrict manager SNMP views to access only the needed set of MIBs:** By default, there is no SNMP view entry. If you have any SNMP view on certain MIB trees, every other tree is implicitly denied.
- **Configure ACLs to restrict SNMP access to only known managers:** Access lists should be used to ensure that only authorized end stations have SNMP access to the network device.
- **Implement security mechanisms:** SNMPv3 is the recommended version as it provides authentication, encryption, and integrity. Be aware that the SNMPv1 and SNMPv2c community strings were not designed as a security mechanism and are transmitted in plaintext. Nevertheless, community strings should not be trivial and should be changed at regular intervals.

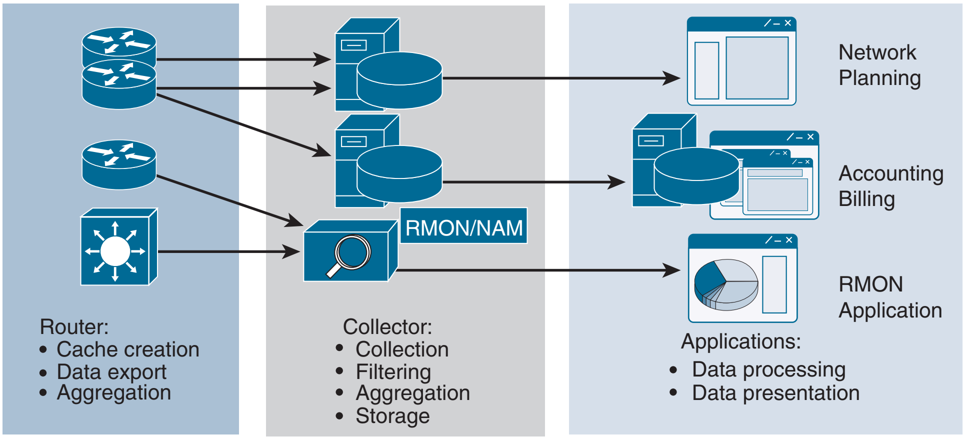
NetFlow

Visibility of network traffic and resource utilization is an important function of network management and capacity planning. Cisco NetFlow is an embedded Cisco IOS Software tool that reports usage statistics for measured resources within a network, giving network managers clear insight about the traffic for analysis.

NetFlow requires three components, as illustrated in Figure 13-3:

- **Flow exporter:** This is a router or another network device that is in charge of collecting flow information and exporting it to a flow collector.
- **Flow collector:** This is a server that receives the exported flow information.
- **Flow analyzer:** This is an application that analyzes flow information collected by the flow collector.

Figure 13-3 NetFlow Process


RMON/NAM
Network Planning
Accounting Billing
RMON Application
Router:
• Cache creation
• Data export
• Aggregation
Collector:
• Collection
• Filtering
• Aggregation
• Storage
Applications:
• Data processing
• Data presentation

Routers and switches that support NetFlow can collect IP traffic statistics on all interfaces where NetFlow is enabled and later export those statistics as NetFlow records toward at least one NetFlow collector—typically a server that does the traffic analysis.

NetFlow facilitates solutions for many common problems encountered by IT professionals, including the following:

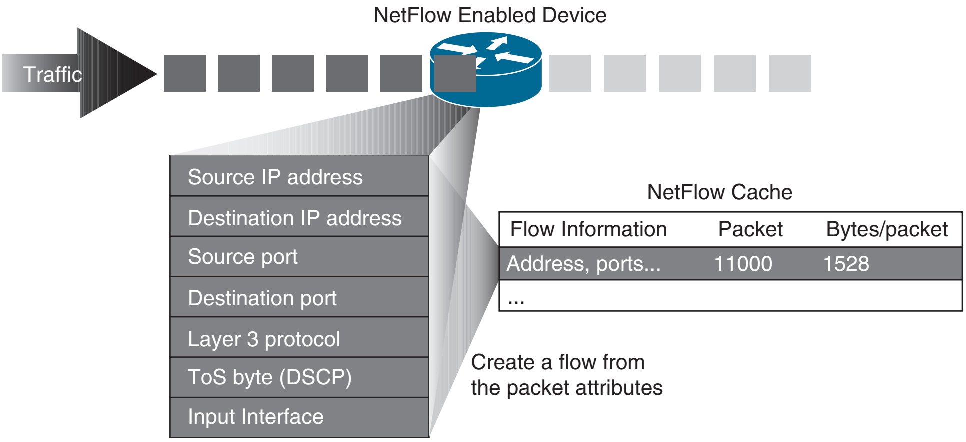
- Analysis of new applications and their impact on the network
- Analysis of WAN traffic statistics
- Troubleshooting and understanding of network challenges
- Detection of unauthorized WAN traffic
- Detection of security and anomalies
- Validation of QoS parameters

Creating a Flow in the NetFlow Cache

NetFlow delivers detailed usage information about IP traffic flows that are traversing a device such as a Cisco router. An IP traffic flow can be described as a stream of packets that is related to the same conversation between two devices.

NetFlow identifies a traffic flow by identifying several characteristics within the packet header, such as source and destination IP addresses, source and destination ports, and Differentiated Services Code Point (DSCP) or ToS markings, as illustrated in Figure 13-4. Once the traffic flow is identified, subsequent packets that match those attributes are regarded as part of that flow.

Figure 13-4 NetFlow Packet Attributes


NetFlow Enabled Device
Traffic
Source IP address
Destination IP address
Source port
Destination port
Layer 3 protocol
ToS byte (DSCP)
Input Interface
NetFlow Cache
Create a flow from
the packet attributes

Flow Information	Packet	Bytes/packet
Address, ports...	11000	1528
...		

Each packet that is forwarded within a router or switch is examined for a set of IP packet attributes. These attributes are the IP packet identity or fingerprint of the packet, and they determine whether the packet is unique or similar to other packets.

An IP flow is based on a set of five to seven IP packet attributes:

- IP source address
- IP destination address
- Source port
- Destination port
- Layer 3 protocol type
- ToS (DSCP)
- Router or switch interface

All packets with the same source and destination IP addresses, source and destination ports, protocol interface, and ToS/DSCP are grouped into a flow, and then packets and bytes are tallied. This methodology of fingerprinting, or determining a flow, is scalable because a large amount of network information is condensed into a database of NetFlow information that is called the NetFlow cache. This flow information is useful for understanding network behavior and usage characteristics. The source address indicates who originated the traffic. The destination address tells who is receiving the traffic. The ports characterize the application utilizing the traffic. The ToS/DSCP indicates the priority of the traffic. The device interface tells how traffic is used by the network device. Tallied packets and bytes show the amount of traffic.

NetFlow Data Analysis

The flow data that is collected in the NetFlow cache is useless unless an administrator can access it. There are two primary methods of accessing NetFlow data: using the CLI with Cisco IOS Software **show** commands or using an application reporting tool called a NetFlow collector.

If you want an immediate view of what is happening in your network, you can use the CLI. The CLI commands can yield onscreen output of the cached data store and can be filtered to produce more specific output. The NetFlow CLI is useful for troubleshooting and real-time analysis of traffic utilization. From a security standpoint, this real-time information is critical to detecting anomalous behavior in the traffic stream.

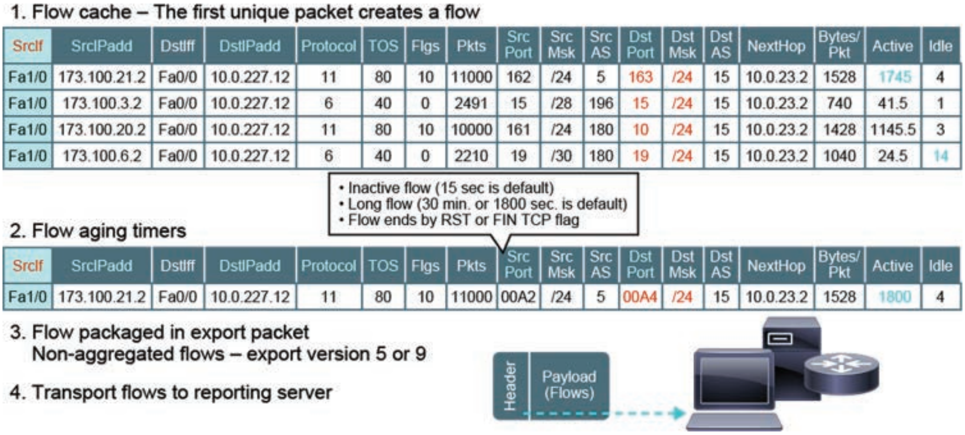
A NetFlow collector can assemble exported flows and then combine or aggregate them to produce reports that can be used for traffic and security analysis. A NetFlow export, unlike SNMP polling, pushes information periodically to the NetFlow collector. Typically, the NetFlow cache constantly fills with flows, and software in the router or switch searches the cache for flows that have terminated or expired. These flows are exported to the NetFlow collector server. Flows are terminated when the network communication has ended (for example, when a packet contains the TCP FIN flag).

Once the NetFlow data is collected and cached, the switch or router must determine which flows to export to the NetFlow collector. A network administrator can use the NetFlow monitor to associate various records to the configured exporters. There can be multiple NetFlow collectors in a network, and a network administrator can send specific NetFlow record data to one or more of those collectors, if necessary.

A flow is ready for export when it is inactive for a certain time (that is, no new packets are received for the flow) or when the flow is long-lived (active) and lasts longer than the active timer (for example, a long FTP download). A flow is also ready for export when a TCP flag (such as FIN or RST) indicates that the flow is terminated. Timers can determine whether a flow is inactive or if a flow is live. The default for the inactive flow timer is 15 seconds, and the default for the active flow timer is 30 minutes, but all timers for export are configurable.

A NetFlow collector can combine flows and aggregate traffic. For example, an FTP download that lasts longer than the active timer may be broken into multiple flows, and the collector can combine these flows to show the total FTP traffic to a server at a specific time of day. Figure 13-5 illustrates this process.

Figure 13-5 NetFlow Packet Format and Flow Transmission



1. Flow cache – The first unique packet creates a flow

SrcIf	SrcIPadd	DstIf	DstIPadd	Protocol	TOS	Flgs	Pkts	Src Port	Src Msk	Src AS	Dst Port	Dst Msk	Dst AS	NextHop	Bytes/ Pkt	Active	Idle
Fa1/0	173.100.21.2	Fa0/0	10.0.227.12	11	80	10	11000	162	/24	5	163	/24	15	10.0.23.2	1528	1745	4
Fa1/0	173.100.3.2	Fa0/0	10.0.227.12	6	40	0	2491	15	/28	196	15	/24	15	10.0.23.2	740	41.5	1
Fa1/0	173.100.20.2	Fa0/0	10.0.227.12	11	80	10	10000	161	/24	180	10	/24	15	10.0.23.2	1428	1145.5	3
Fa1/0	173.100.6.2	Fa0/0	10.0.227.12	6	40	0	2210	19	/30	180	19	/24	15	10.0.23.2	1040	24.5	14

- Inactive flow (15 sec is default)
- Long flow (30 min. or 1800 sec. is default)
- Flow ends by RST or FIN TCP flag

2. Flow aging timers

SrcIf	SrcIPadd	DstIf	DstIPadd	Protocol	TOS	Flgs	Pkts	Src Port	Src Msk	Src AS	Dst Port	Dst Msk	Dst AS	NextHop	Bytes/ Pkt	Active	Idle
Fa1/0	173.100.21.2	Fa0/0	10.0.227.12	11	80	10	11000	00A2	/24	5	00A4	/24	15	10.0.23.2	1528	1800	4

3. Flow packaged in export packet
   Non-aggregated flows – export version 5 or 9

4. Transport flows to reporting server

NetFlow Export Data Format

The format of the export data depends on the version of NetFlow that is used in the network architecture. There are various formats for export packets, depending on the export version.

The differences between the versions of NetFlow are evident in the version-dependent packet header fields. The export versions, including Versions 5, 7, and 9, are well-documented formats. In the past, the most commonly used format was NetFlow Version 5, but Version 9 is the latest format and has some advantages for key technologies such as security, traffic analysis, and multicast.

The NetFlow Version 9 data export format is a flexible and extensible format, which provides the versatility needed for support of new fields and record types. The main feature of NetFlow Version 9 export format is that it is template based. A template describes a NetFlow record format and attributes of fields (such as type and length) within a record. The router assigns each template an ID, which is communicated to the NetFlow collector, along with the template description. The template ID is used for all further communication from the router to the NetFlow collector.

These templates allow the NetFlow Version 9 data export format to accommodate NetFlow-supported technologies such as multicast, Multiprotocol Label Switching (MPLS), and Border Gateway Protocol (BGP) next hop. The Version 9 export format enables you to use the same version for main and aggregation caches, and the format is extendable, so you can use the same export format with future features.

There is also a Version 10 data export format, but this version is used for identifying IPFIX. Although IPFIX is heavily based on NetFlow, Version 10 does not have anything to do with NetFlow, and the NetFlow protocol itself has been replaced by IPFIX. Based on the NetFlow Version 9 implementation, IPFIX is on the IETF standards track with RFC 5101 (made obsolete by RFC 7011), RFC 5102 (made obsolete by RFC 7012), and other RFCs published in 2008.

Traditional NetFlow Configuration and Verification

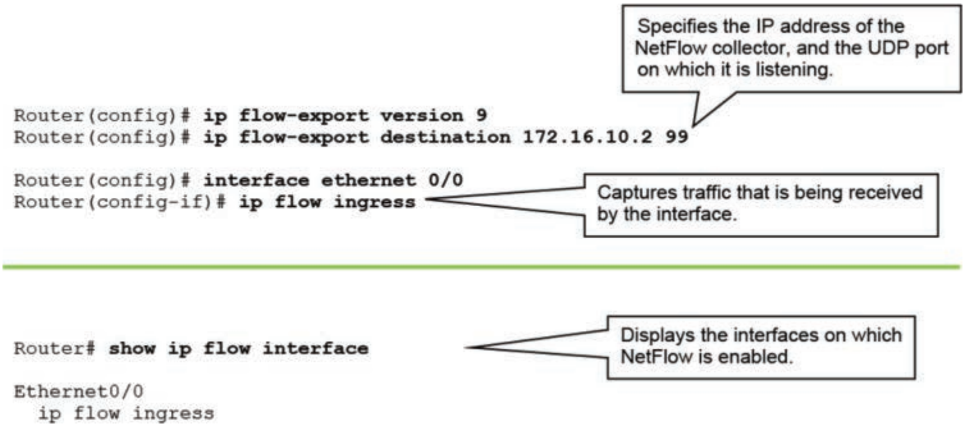
Figure 13-6 shows the commands to configure and verify traditional NetFlow Version 9.

In this example, the NetFlow collector is located at IP address 172.16.10.2, and it is listening on UDP port 99. Also, data is being collected about traffic entering interface Ethernet 0/0 on the router. You can configure NetFlow to capture flows for traffic transmitted out an interface as well. The Egress NetFlow Accounting feature captures NetFlow statistics for IP traffic only. MPLS statistics are not captured. However, the MPLS Egress NetFlow Accounting feature can be used on a provider edge (PE) router to capture IP traffic flow information for egress IP packets that arrived at the router as MPLS packets and underwent label disposition.

Egress NetFlow accounting might adversely affect network performance because of the additional accounting-related computation that occurs in the traffic-forwarding path of the router.

Also, note that NetFlow consumes additional memory. If you have memory constraints, you might want to preset the size of the NetFlow cache so that it contains a smaller number of entries. The default cache size depends on the platform. NetFlow Version 9 is not backward compatible with Version 5 or Version 8. If you need Version 5 or Version 8, you must configure it.

Figure 13-6 Traditional NetFlow Version 9 Configuration



Specifies the IP address of the NetFlow collector, and the UDP port on which it is listening.

```
Router(config)# ip flow-export version 9
Router(config)# ip flow-export destination 172.16.10.2 99

Router(config)# interface ethernet 0/0
Router(config-if)# ip flow ingress
```

Captures traffic that is being received by the interface.

```
Router# show ip flow interface

Ethernet0/0
  ip flow ingress
```

Displays the interfaces on which NetFlow is enabled.

To verify the traffic flows that NetFlow is capturing, use the **show ip cache flow** command, as illustrated in Example 13-1.

Example 13-1 Verifying NetFlow Data

```
Router# show ip cache flow
IP packet size distribution (1103746 total packets):
   1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
   .249 .694 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000 .000
    512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
   .000 .000 .027 .000 .027 .000 .000 .000 .000 .000 .000

IP Flow Switching Cache, 278544 bytes
  35 active, 4061 inactive, 980 added
  2921778 ager polls, 0 flow alloc failures
  Active flows timeout in 30 minutes
  Inactive flows timeout in 15 seconds
IP Sub Flow Cache, 21640 bytes
  0 active, 1024 inactive, 0 added, 0 added to flow
  0 alloc failures, 0 force free
  1 chunk, 1 chunk added
  last clearing of statistics never
Protocol          Total    Flows   Packets  Bytes  Packets Active(Sec) Idle(Sec)
--------          Flows     /Sec     /Flow   /Pkt     /Sec     /Flow      /Flow
TCP-FTP             108      0.0      1133     40      2.4    1799.6        0.9
TCP-FTPD            108      0.0      1133     40      2.4    1799.6        0.9
TCP-WWW              54      0.0      1133     40      1.2    1799.6        0.8
TCP-SMTP             54      0.0      1133     40      1.2    1799.6        0.8
TCP-BGP              27      0.0      1133     40      0.6    1799.6        0.7
TCP-NNTP             27      0.0      1133     40      0.6    1799.6        0.7
TCP-other           297      0.0      1133     40      6.8    1799.7        0.8
UDP-TFTP             27      0.0      1133     28      0.6    1799.6        1.0
UDP-other           108      0.0      1417     28      3.1    1799.6        0.9
ICMP                135      0.0      1133    427      3.1    1799.6        0.8
Total:              945      0.0      1166     91     22.4    1799.6        0.8

SrcIf          SrcIPaddress    DstIf          DstIPaddress    Pr SrcP DstP  Pkts
Et0/0          192.168.67.6    Et1/0.1        172.16.10.200   01 0000 0C01    51
Et0/0          10.10.18.1      Null           172.16.11.5     11 0043 0043    51
Et0/0          10.10.18.1      Null           172.16.11.5     11 0045 0045    51
Et0/0          10.234.53.1     Et1/0.1        172.16.10.2     01 0000 0800    51
Et0/0          10.10.19.1      Null           172.16.11.6     11 0044 0044    51
Et0/0          10.10.19.1      Null           172.16.11.6     11 00A2 00A2    51
Et0/0          192.168.87.200  Et1/0.1        172.16.10.2     06 0014 0014    50
Et0/0          192.168.87.200  Et1/0.1        172.16.10.2     06 0015 0015    52
<. . . output omitted . . .>
```

In this output, you can see that there are currently 35 active flows; the most popular of these flows are listed in the Protocol column.

Flexible NetFlow

Flexible NetFlow is an extension of NetFlow Version 9 that provides additional functionality, enabling you to export more information by using the same NetFlow Version 9 datagram. Flexible NetFlow provides flexibility and scalability of flow data beyond traditional NetFlow.

Flexible NetFlow allows you to understand network behavior with more efficiency, with specific flow information tailored for various services used in the network. It enhances Cisco NetFlow as a security monitoring tool. For instance, new flow keys can be defined for packet length or MAC address to allow users to search for a specific type of attack in a network. Flexible NetFlow allows you to quickly identify how much application traffic is being sent between hosts by specifically tracking TCP or UDP applications based on the Type of Service (ToS) field in the packets. Flexible NetFlow provides accounting of traffic entering a Multiprotocol Label Switching (MPLS) or IP core network, as well as its destination for each next hop per class of service. This makes it possible to build an edge-to-edge traffic matrix.

Traditional vs. Flexible NetFlow

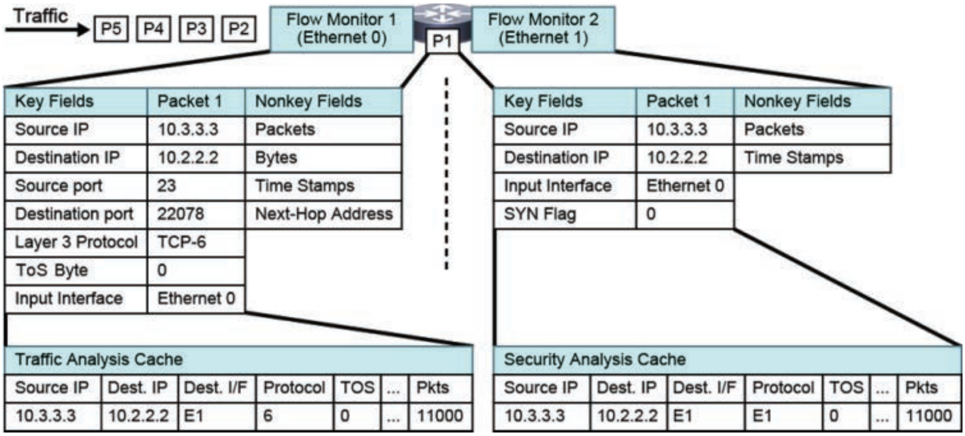
NetFlow and Flexible NetFlow both use the values in key fields in IP datagrams—such as the IP source or destination address and the source or destination transport protocol port—as criteria for determining when a new flow must be created in the cache while network traffic is being monitored. When the value of the data in the key field of a datagram is unique with respect to the flows that exist, a new flow is created. Traditional NetFlow is based on a set of up to seven IP packet attributes. Flexible NetFlow allows a flow to be user defined; a user can configure key fields for detailed traffic analysis.

Whereas NetFlow has a single cache, and all applications use the same cache information, Flexible NetFlow has the capability to create multiple flow caches or information databases to track NetFlow information. Flexible NetFlow applications such as security monitoring, traffic analysis, and billing can be tracked separately, and the information can be customized for each application. Each cache has the specific and customized information required for the application. For example, multicast and security information can be tracked separately, and the results can be sent to two different NetFlow reporting systems.

With NetFlow, typically up to seven IP packet fields are tracked to create NetFlow information, and the fields used to create the flow information are not configurable. With Flexible NetFlow, the user configures what to track, which means fewer flows are produced; this increases the scalability of hardware and software resources. For example, IPv4 header information, BGP information, and multicast or IPv6 data can all be configured and tracked in Flexible NetFlow.

NetFlow typically tracks IP information such as IP addresses, ports, protocols, and TCP flags, and most security systems look for anomalies or changes in network behavior to detect security incidents. Flexible NetFlow allows the user to track a wide range of IP information, including all the fields in the IPv4 header or IPv6 header and various individual TCP flags, and it can also export sections of a packet. The information being tracked may be a key field (used to create a flow) or a non-key field (collected with the flow). The user has the ability to use one NetFlow cache to detect security vulnerabilities (anomaly detection) and create a second cache to focus on a particular problem. Figure 13-7 shows an example of this process in which a packet is analyzed by two different NetFlow monitor functions on the router. Flow monitor 1 builds a traffic analysis cache, and flow monitor 2 builds a security analysis cache.

Figure 13-7 Cisco Flexible NetFlow Cache



Key Fields	Packet 1	Nonkey Fields
Source IP	10.3.3.3	Packets
Destination IP	10.2.2.2	Bytes
Source port	23	Time Stamps
Destination port	22078	Next-Hop Address
Layer 3 Protocol	TCP-6	
ToS Byte	0	
Input Interface	Ethernet 0	

Traffic Analysis Cache

Source IP	Dest. IP	Dest. I/F	Protocol	TOS	...	Pkts
10.3.3.3	10.2.2.2	E1	6	0	...	11000

Key Fields	Packet 1	Nonkey Fields
Source IP	10.3.3.3	Packets
Destination IP	10.2.2.2	Time Stamps
Input Interface	Ethernet 0	
SYN Flag	0	

Security Analysis Cache

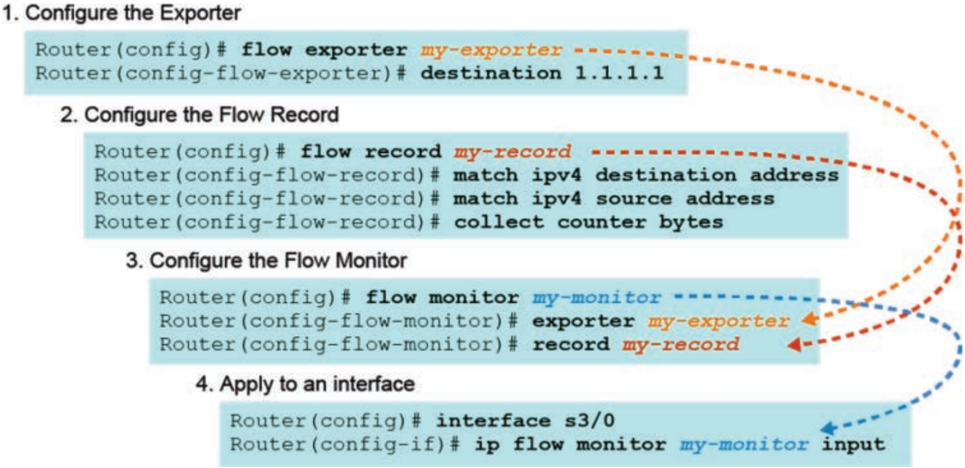
Source IP	Dest. IP	Dest. I/F	Protocol	TOS	...	Pkts
10.3.3.3	10.2.2.2	E1	E1	0	...	11000

Within Cisco DNA Center, Flexible NetFlow and Application Visibility and Control (AVC) with NBAR2 are leveraged by the Cisco DNA Center Analytics engine to provide context when troubleshooting poor user experience.

Flexible NetFlow Configuration and Verification

Figure 13-8 illustrates the four basic steps required to configure Cisco Flexible NetFlow.

Figure 13-8 Cisco Flexible NetFlow Configuration Steps



1. Configure the Exporter

```
Router(config)# flow exporter my-exporter
Router(config-flow-exporter)# destination 1.1.1.1
```

2. Configure the Flow Record

```
Router(config)# flow record my-record
Router(config-flow-record)# match ipv4 destination address
Router(config-flow-record)# match ipv4 source address
Router(config-flow-record)# collect counter bytes
```

3. Configure the Flow Monitor

```
Router(config)# flow monitor my-monitor
Router(config-flow-monitor)# exporter my-exporter
Router(config-flow-monitor)# record my-record
```

4. Apply to an interface

```
Router(config)# interface s3/0
Router(config-if)# ip flow monitor my-monitor input
```

The first step is to configure a Flexible NetFlow exporter to describe where the flows are sent. This terminology is confusing because most NetFlow users (including the Stealthwatch system) consider an "exporter" to be the router. From the router's perspective, however, the exporter is the device that information is being exported to. When configuring the exporter, you can optionally specify a source interface and a UDP port number to use for transmission to the collector.

The second step is to define a flow record. A NetFlow record is a combination of key and non-key fields used to identify flows. Both predefined and user-defined records can be configured. Customized user-defined flow records are used to analyze traffic data for a specific purpose.

A customized flow record must have at least one **match** criterion for use as the key field and typically has at least one **collect** criterion for use as a non-key field. You have to specify a series of **match** and **collect** commands to tell the router which fields to include in the outgoing NetFlow PDU. The **match** fields are the key fields: They are used to determine the uniqueness of the flow. The **collect** fields are just extra information (non-key) that you can include to provide more detail to the collector for reporting and analysis. A best practice is to use **match** for all seven key fields (Source IP Address, Destination IP Address, Source Port, Destination Port, Input Interface, Layer 3 Protocol, and ToS). You can then use **collect** with optional fields, such as Counters, Timestamps, Output Interface, and DSCP.

The third step is to configure a flow monitor, which represents the memory-resident NetFlow database of the router. Flexible NetFlow allows you to create multiple independent monitors. While this can be useful in some situations, most users create a single main cache for collecting and exporting NetFlow data. This step binds together the flow exporter and the flow record. You can optionally change the default cache timeout values.

The last step is to apply the flow monitor to each Layer 3 interface on the router. Flexible NetFlow should be enabled at each entry point to the router. In almost all cases, you want to use input monitoring.

You can use the **show flow monitor** and **show flow monitor cache** commands to verify the Flexible NetFlow process, as shown in Example 13-2.

Example 13-2 Verifying Flexible NetFlow

```
Router# show flow monitor
Flow Monitor my-monitor:
  Description:        Main Cache
  Flow Record:        my-record
  Flow Exporter:      my-exporter
  Cache:
    Type:                 normal
    Status:               allocated
    Size:                 4096 entries / 311316 bytes
    Inactive Timeout:     15 secs
    Active Timeout:       1800 secs
    Update Timeout:       1800 secs

Router# show flow monitor my-monitor cache
  Cache type:                                   Normal
  Cache size:                                     4096
```

```
   Current entries:                                  5
   High Watermark:                                   6

   Flows added:                                     62
   Flows aged:                                      57
     - Active timeout       (     60 secs)          57
     - Inactive timeout     (     60 secs)           0
     - Event aged                                    0
     - Watermark aged                                0
     - Emergency aged                                0

IPV4 SOURCE ADDRESS:        10.10.10.10
IPV4 DESTINATION ADDRESS:   10.20.20.10
counter bytes:              500
```

In this output, notice the captured fields of information that match the flow record configured in Figure 13-8.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	24
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	11

This page intentionally left blank

Day 12

Wireless Concepts

ENCOR 350-401 Exam Topics

Infrastructure

- Wireless
  - Describe Layer 1 concepts, such as RF power, RSSI, SNR, interference noise, band and channels, and wireless client devices capabilities

Key Topics

Today we start reviewing wireless concepts. Over the next three days, we cover wireless principles, deployment options, roaming and location services, and access point (AP) operation and client authentication.

To fully understand Wi-Fi technology, you must have a clear concept of how Wi-Fi fundamentally works. Today we explore Layer 1 concepts of radio frequency (RF) communications, the types of antennas used in wireless communication, and the Institute of Electrical and Electronics Engineers (IEEE) 802.11 standards that wireless clients must comply with to communicate over radio frequencies. We also look at the functions of different components of an enterprise wireless solution.

Explain RF Principles

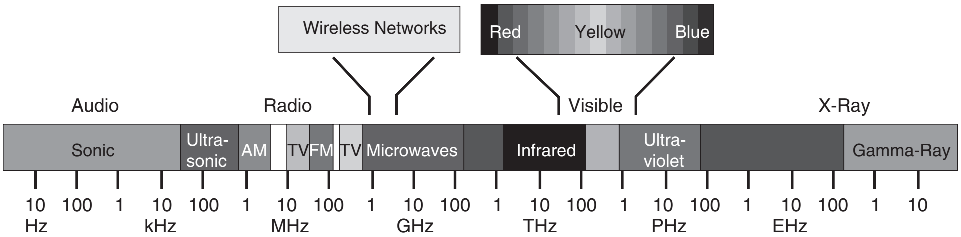
Radio frequency (RF) communications are at the heart of the wireless physical layer. This section gives you the tools you need to understand the use of RF waves as a means of transmitting information

RF Spectrum

Many devices use radio waves to send information. A radio wave is an electromagnetic field (EMF) that radiates from a transmitter. This wave propagates to a receiver, which receives the energy. Light is an example of electromagnetic energy. The human eye can interpret light and send the light energy to the brain, which transforms this energy into impressions of colors.

Different waves have different sizes, typically expressed in meters. Another unit of measurement, hertz, expresses how often a wave occurs per second (see Figure 12-1). Waves are grouped by category, with each group matching a size variation. The highest waves are in the gamma-ray group.

Figure 12-1 Continuous Frequency Spectrum


Wireless Networks
Red
Yellow
Blue
Audio
Radio
Visible
X-Ray
Sonic
Ultra-sonic
AM
TV
FM
TV
Microwaves
Infrared
Ultra-violet
Gamma-Ray
10 Hz
100
1
10 kHz
100
1
10 MHz
100
1
10 GHz
100
1
10 THz
100
1
10 PHz
100
1
10 EHz
100
1
10

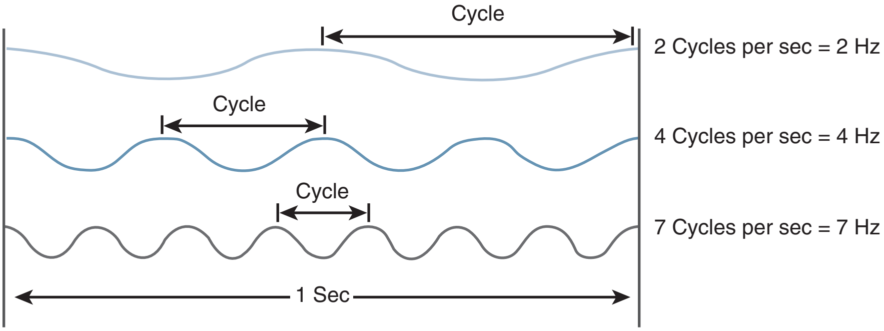
The waves on the spectrum that a human body cannot perceive are used to send information. Depending on the type of information being sent, certain wave groups are more efficient than others in the air because they have different properties. In wireless networks, because of the different needs and regulations that have arisen over time, it has become necessary to create subgroups.

Frequency

A wave is always sent at the speed of light because it is an electromagnetic field. Therefore, the amount of time a wave takes to travel one cycle depends on the length of the wave.

For example, a signal wavelength that is 0.2 inch (5 mm) long takes less time to travel a cycle than does a signal wavelength that is 1312 feet (400 m) long. The speed is the same in both cases, but because a longer signal takes more time to travel one cycle than a shorter signal, the longer signal goes through fewer cycles in 1 second than does the shorter signal. This principle is illustrated in Figure 12-2, where you can see that a 7 Hz signal repeats more often in one second than does a 2 Hz signal.

Figure 12-2 Cycles Within a Wave


Cycle
2 Cycles per sec = 2 Hz
Cycle
4 Cycles per sec = 4 Hz
Cycle
7 Cycles per sec = 7 Hz
1 Sec

A direct relationship exists between the frequency of a signal (how often the signal is seen) and the wavelength of the signal (the distance that the signal travels in one cycle). The shorter the wavelength, the more often the signal repeats over a given amount of time and, therefore, the higher the frequency.

A signal that occurs 1 million times per second is a megahertz, and a signal that occurs 1 billion times per second is a gigahertz. This fact is important in Wi-Fi networks because lower-frequency signals are less affected by the air than are higher-frequency signals.

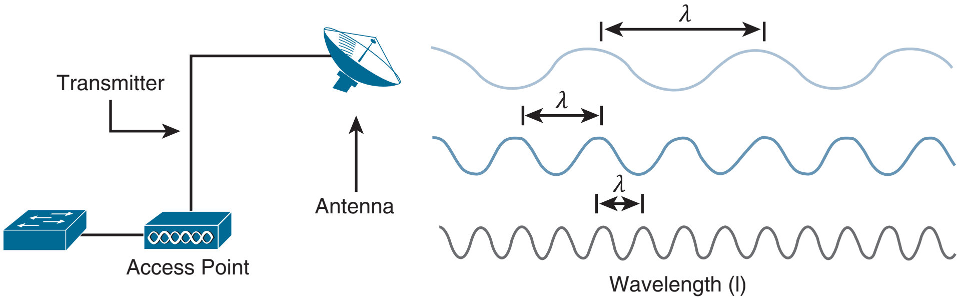
Wavelength

An RF signal starts with an electrical alternating current (AC) signal that a transmitter generates. This signal is sent through a cable to an antenna, which radiates the signal in the form of an electromagnetic wireless signal. Changes of electron flow in the antenna, known as *current*, produce changes in the electromagnetic fields around the antenna and transmit electric and magnetic fields.

AC is electrical current in which the direction of the current changes cyclically. The shape and form of an AC signal—the *waveform*—is known as a sine wave. This shape is the same as the signal that the antenna radiates on a wireless access point.

The physical distance from one point of a wave's cycle to the same point in the next cycle is called a *wavelength* and is typically represented by the Greek symbol lambda (λ). The wavelength is the physical distance that the wave covers in one cycle. This is illustrated in Figure 12-3, where the waves are arranged in order of increasing frequency, from top to bottom. Notice that the wavelength decreases as the frequency increases.

Figure 12-3 Wireless Signal Transmission, with Examples of Increasing Frequency and Decreasing Wavelength



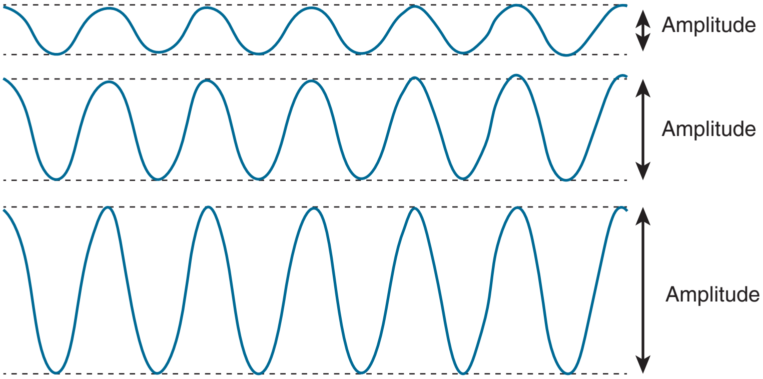
Wavelength distance determines some important properties of the wave. Certain environments and obstacles can affect the wave, and the degree of impact varies depending on the wavelength and the obstacle that the wave encounters. This phenomenon is covered in more detail later today.

Some AM radio stations use a wavelength that is 1312 or 1640 feet (400 or 500 m) long. Wi-Fi networks use a wavelength that is a few centimeters long. Some satellites use wavelengths that are about 0.04 inch (1 mm) long.

Amplitude

Amplitude is another important factor that affects how a wave is sent. Amplitude is the strength of a signal. In a graphical representation, amplitude is the distance between the highest and lowest crests of the cycle, as illustrated in Figure 12-4.

Figure 12-4 Signal Amplitude



The Greek symbol gamma (γ) is commonly used to represent amplitude. Amplitude affects the signal because it represents the level of energy that is injected in one cycle. The more energy that is injected in a cycle, the higher the amplitude.

Amplification is the increase of the amplitude of a wave. Amplification can be active or passive. In active amplification, the applied power is increased. Passive amplification is accomplished by focusing the energy in one direction by using an antenna. Amplitude can also be decreased. A decrease in amplitude is called *attenuation*.

Finding the right amplitude for a signal can be difficult. A signal weakens as it moves away from the emitter. If a signal is too weak, it might be unreadable when it arrives at the receiver. If a signal is too strong, then generating it requires too much energy (making the signal costly to generate). In addition, high signal strength can damage a receiver.

Regulations exist for determining the right amount of power to use for each type of device, depending on the expected distance over which the signal will be sent. Following these regulations helps avoid problems related to using the wrong amplitude.

Free Path Loss

A radio wave that an access point (AP) emits is radiated in the air. With an omnidirectional antenna, the signal is emitted in all directions—as when a stone is thrown into water, and waves radiate outward from the point at which the stone touches the water. If an AP uses a directional antenna, the beam is more focused in one direction.

As the signal or wave travels away from the AP, it is affected by any obstacles that it encounters. The exact effect depends on the type of obstacle the wave encounters. Even without encountering any obstacle, the first effect of wave propagation is strength attenuation.

Continuing with the example of a stone being thrown into water, the generated radio wave circles have higher crests close to the center than they do farther out. As the distance increases, the circles become flatter, until they finally disappear completely.

The attenuation of signal strength on its way between a sender and a receiver is called *free path loss*, or *free space path loss*. The word *free* in this case refers to the fact that the loss of energy is simply a result of distance; that is, it is not due to any obstacle. *Path loss*, on the other hand, also considers other sources of loss.

Keep in mind that what causes free path loss is not distance itself; there is no physical reason for a signal to be weaker farther away from the source. The cause of the loss is actually the combination of two phenomena:

- The signal is sent from the emitter in all directions. The energy must be distributed over a larger area (a larger circle), but the amount of energy that is originally sent does not change. Therefore, the amount of energy that is available on each point of the circle is higher if the circle is small (with fewer points) than if the circle is large (with more points among which the energy must be divided).
- The receiver antenna has a certain physical size, and the amount of energy that is collected depends on this size. A large antenna collects more points of the circle than a small one. But regardless of size, an antenna cannot pick up more than a portion of the original signal, especially because this process occurs in three dimensions (whereas the stone in water example occurs in two dimensions); the rest of the sent energy is lost.

The combination of these two factors causes free path loss. If energy could be emitted toward a single direction and if the receiver could catch 100% of the sent signal, there would be no loss at any distance because there would be nothing along the path to absorb any signal strength.

Some antennas are built to focus the signal as much as possible to try to send a powerful signal far from the AP. But the focus is still not perfect, so receivers cannot capture 100% of what is sent.

RSSI and SNR

Because an RF wave can be affected by obstacles in its path, it is important to determine how much signal the other endpoint will receive. The signal can become too weak for the receiver to hear or detect it as a signal. Two values are important in this determination: RSSI and SNR.

RSSI

Received signal strength indicator (*RSSI*), also called the *signal value*, indicates how much power is received. RSSI is the strength of a signal that one device receives from another device. RSSI is usually expressed in decibels referenced to 1 milliwatt (dBm).

Calculating RSSI is a complex problem because the receiver does not know how much power was originally sent. RSSI expresses a relative value that the receiving wireless network card determines while comparing received packets to each other.

RSSI is a grade value that can range from 0 (no signal or no reference) to a maximum of 255. However, many vendors use a maximum value that is lower than 255 (for example, 100 or 60). The value is relative because a magnetic field and an electric field are received, and a transistor transforms them into electric power; current is not directly received. How much electric power can be generated depends on the received field and the circuit that transforms it into current.

From the RSSI grade value, an equivalent dBm value is displayed, and this value depends on the vendor. One vendor might determine that the RSSI for a card will range from 0 to 100, where 0 is represented as –95 dBm and 100 as –15 dBm; another vendor might determine that the range will be 0 to 60, where 0 is represented as –92 dBm and 60 as –12 dBm. In this case, it is not possible to compare powers when reading RSSI = –35 dBm on the first product and RSSI = –28 dBm on the second product. For Cisco products, good RSSI values would be –67 dBm or better (for example, –55 dBm).

Therefore, RSSI is not a means of comparing wireless network cards; rather, it is a way to help understand, card by card, how strong a received signal is relative to itself in different locations. This method is useful for troubleshooting or when comparing the values of cards by the same vendor.

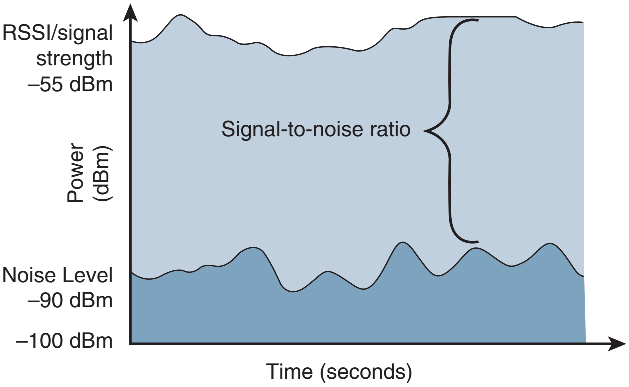
An attempt is being made to unify these values through the received channel power indicator (RCPI). Future cards might use the RCPI, which will be the same scale on all cards, instead of RSSI.

Noise (or noise floor) can be caused by wireless devices, such as cordless phones and microwaves. The noise value is measured in decibels, from 0 to –120. The noise level is the amount of interference in the Wi-Fi signal, and the lower value, the better. A typical noise floor value would be –95 dBm.

SNR

Another important metric is signal-to-noise ratio (SNR). SNR is a ratio value that evaluates a signal, based on the noise present. SNR is measured as a positive value between 0 and 120; the closer the value is to 120, the better.

Figure 12-5 SNR Example



SNR comprises two values, as shown in Figure 12-5:

- RSSI
- Noise (any signal that interferes with the signal)

To calculate the SNR value, subtract the noise value from the RSSI. Because both values are usually expressed as negative numbers, the result is a positive number that is expressed in decibels. For example, if the RSSI is –55 dBm and the noise value is –90 dBm, you can make the following calculation:

$$-55 \text{ dBm} - (-90 \text{ dBm}) = -55 \text{ dBm} + 90 \text{ dBm} = 45 \text{ dB}$$

So in this case, you have an SNR of 45 dB. The general principle is that any SNR above 20 dB is good. These values depend not only on the background noise but also on the speed that is to be achieved.

As an example of SNR in everyday life, when someone speaks in a room, a certain volume is enough to be heard and understood. But if the same person speaks outdoors, surrounded by the noise of traffic, the same volume might be enough to be heard but not enough to be understood. In a very quiet room, a whisper can be heard. Although the voice is almost inaudible, it is easy to understand because it is the only sound present. In a noisy outdoor environment, isolating the voice from the surrounding noise is more difficult, and the voice needs to be much louder than the surrounding noise to be understood.

Current calculations use the signal-to-interference-plus-noise ratio (SINR). This calculation takes into account the noise floor and the strength of any interference to the signal. An SINR calculation is the RSSI minus the combination of interference and noise. An SINR of 25 or better is required for voice over wireless LAN (VoWLAN) applications.

Watts and Decibels

A key problem in Wi-Fi network design is determining how much power is or should be sent from a source and, therefore, is or should be received by the endpoint. The distances that can be achieved depend on this determination. The power that is sent from a source also determines which device to install, the type of AP to use, and the type of antenna to use.

The first unit of power that is used in power measurement is the watt (W), which is named after James Watt. The watt is a measure of the energy spent (that is, emitted or consumed) per second; 1 W represents 1 joule (J) of energy per second. A joule is the amount of energy that is generated by a force of 1 newton (N) moving 1 meter in one direction. A newton is the force that is required to accelerate 1 kilogram (kg) at a rate of 1 meter per second squared (m/s^2).

Watt or milliwatt is an absolute power value that simply expresses power consumption. These measurements are also useful in comparing devices. For example, a typical AP can have a power of 100 mW, but this power varies depending on the context (indoor or outdoor) and the country because there are some regulations in this field.

Another value that is commonly used in Wi-Fi networks is the decibel (dB), which is used to indicate sound levels. A decibel is a logarithmic unit of measurement that expresses the amount of power relative to a reference.

Calculating decibels can be more challenging than simply understanding them. To simplify the task, remember these main values:

- **10 dB:** When the power is 10 dB, the compared value is 10 times more powerful than the reference value. This process also works around the other way: If the compared value is 10 times less powerful than the reference value, then the compared value is written as –10 dB.
- **3 dB:** Remember that decibels are logarithmic. If the power is 3 dB, then the compared value is twice as powerful as the reference value. By the same logic, if the compared value is half as powerful as the reference value, then the compared value is written as –3 dB.

Decibels are used extensively in Wi-Fi networks to compare powers. Two types of powers can be compared:

- The electric power of a transmitter
- The electromagnetic power of an antenna

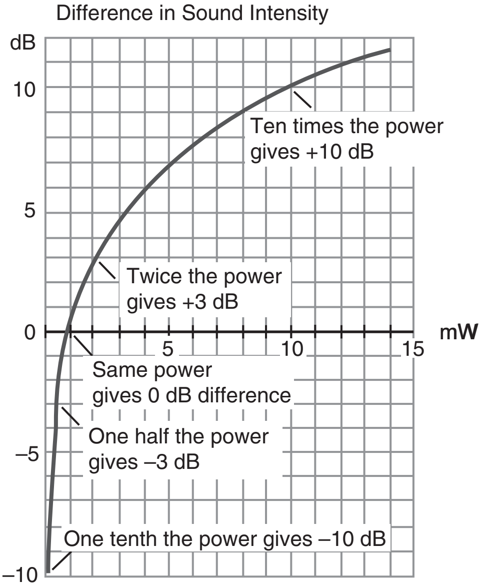
Since the signal that a transmitter emits is an AC current, the power levels are expressed in milliwatts. Comparing powers between transmitters involves comparing values in milliwatts and using dBm as the units.

Following the rules regarding decibels and keeping in mind that a decibel expresses a relative value, you can establish these facts:

- A device that sends at 0 dBm sends the same number of milliwatts as the reference source. If the power reference is 1 mW, the device sends 1 mW.
- A device that sends at 10 dBm sends 10 times as much power (in milliwatts) than the reference source. If the power reference is 1 mW, the device sends 10 mW.
- A device that sends at –10 dBm is one-tenth as powerful as the reference source. If the power reference is 1 mW, the device sends 0.1 mW.
- A device that sends at 3 dBm is twice as powerful as the reference source and sends 2 mW.
- A device that sends at –3 dBm is half as powerful as the reference source and sends 0.5 mW.

These calculations are illustrated in Figure 12-6.

Figure 12-6 Watts to Decibels



By the same logic, a device that sends 6 dBm is four times as powerful as the reference source: Adding 3 dBm makes the device twice as powerful, and adding another 3 dBm makes it twice as powerful again, for a total of 4 mW.

The rules of 3 and 10 allow you to easily determine the transmit power, based on the gain or loss of decibels:

- +3 dB = power times 2
- –3 dB = power divided by 2

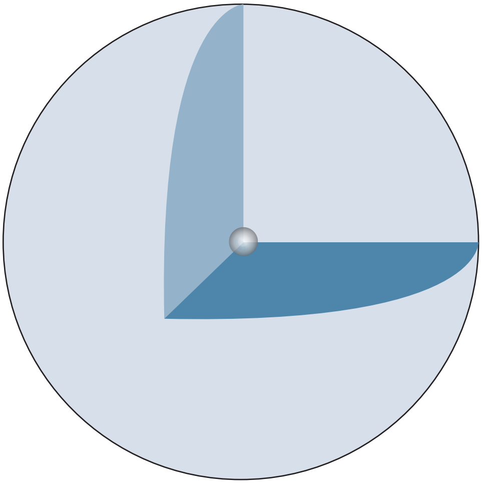
- +10 dB = power times 10
- –10 dB = power divided by 10

So, for every gain of 3 dB, the power is multiplied by 2, and for every gain of 10 dB, the power is multiplied by 10. Conversely, for –3 dB, the power is divided by 2, and for –10 dB, the power is divided by 10. These rules can help you more easily perform calculations of power levels.

Antenna Power

An antenna does not send an electric current; rather, an antenna sends an electromagnetic field. Wi-Fi engineers need to compare the power of antennas without using the indirect value of the current that is sent, and they do so by measuring the power gain relative to a reference antenna.

Figure 12-7 Theoretical Isotropic Antenna



This reference antenna, called an *isotropic antenna*, is a spherical antenna that is theoretically 1 dot large and radiates in all directions, as shown in Figure 12-7. This type of antenna is theoretical and does not exist in reality for two reasons:

- An antenna that is 1 dot large is almost impossible to produce because something would need to be linked to the antenna to send the current to it.
- An antenna usually does not radiate equally in all directions because its construction causes it to send more signal in some directions than in others.

Although this theoretical antenna does not exist, it can be used as a reference to compare actual antennas. The scale that is used to compare the powers that antennas radiate to an isotropic antenna is called dBi (where the *i* stands for *isotropic*).

The logarithmic progression of the dBi scale obeys the same rules as for the other decibel scales:

- 3 dBi is twice as powerful as the theoretical reference antenna.
- 10 dBi is 10 times as powerful as the theoretical reference antenna.

By using the same logarithmic progression, you can compare antennas similarly to the way you compare transmitters. For example, if one antenna is 6 dBi and another is 9 dBi, then the second antenna is 3 dBi more powerful than the first—or two times as powerful.

Other scales can be used to compare antennas. Some Wi-Fi professionals prefer to use a dipole antenna as the reference. This comparison is expressed in dBd. When comparing antennas, be sure to use the same format (either dBd or dBi) for all the antennas involved in the comparison.

Effective Isotropic-Radiated Power

Comparing antennas gives a measure of their gain. An antenna is a passive device, which means it does not add to the energy that it receives from the cable. The only thing that the antenna can do is to radiate the power in one or more directions.

An easy way to understand this concept is to consider a balloon as an example. The quantity of air inside the balloon is the quantity of energy to be radiated. If the balloon is shaped as a sphere, with an imaginary AP at the center, the energy is equally distributed in all directions. The imaginary AP at the center of the balloon radiates in all directions, like an isotropic antenna. Now suppose that the balloon is pressed into the shape of a sausage, and the imaginary AP is placed at one end of this sausage. The quantity of air in the balloon is still the same, but now the energy radiates more in one direction (along the sausage) than in the others.

The same principle applies to antennas. When an antenna concentrates the energy that it receives from the cable in one direction, it is said to be more powerful (in this direction) than an antenna that radiates the energy in all directions because there is more signal in this one direction.

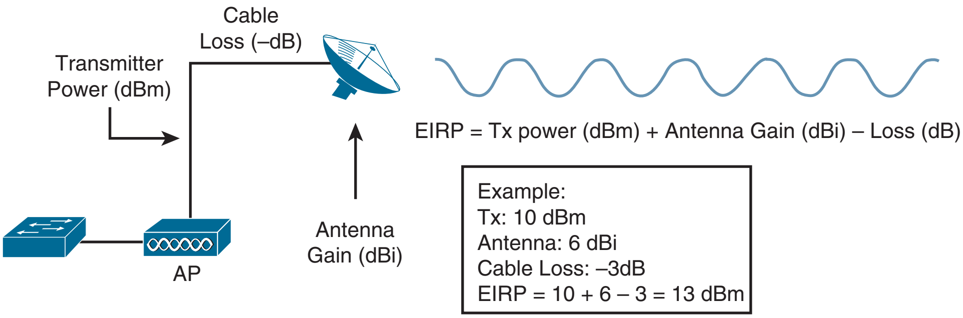
In this sense, describing the power of antennas is like comparing their ability to concentrate the flow of energy in one direction. The more powerful an antenna, the higher its dBi or dBd value, the more it focuses or concentrates the energy that it receives into a narrower beam. But the total amount of power that is radiated is no higher; the antenna does not actively add power to what it receives from the transmitter.

Nevertheless, in the direction toward which the beam is concentrated, the received energy is higher because the receiver gets a higher percentage of the energy that the transmitter emits. And if the transmitter emits more energy, the result is higher yet.

Wi-Fi engineers need a way to determine how much energy is actually radiated from an antenna toward the main beam. This measure is called effective isotropic-radiated power (EIRP). One important concept to keep in mind is that EIRP is isotropic because it is the amount of power that an isotropic antenna would need to emit to produce the peak power density observed in the direction of maximum antenna gain. In other words, EIRP expresses, in isotropic equivalents, how much energy is radiated in the beam. Of course, to do so, EIRP takes into consideration the beam shape and strength and the antenna specifications.

In mathematical terms, EIRP, expressed in dBm, is simply the amount of transmit (Tx) power plus the gain (in dBi) of the antenna. However, the signal might go through a cable, and some power might be lost in the cable, so the cable loss must be deducted.

Figure 12-8 EIRP Calculation Example



Therefore, EIRP can be expressed as EIRP = Tx power (dBm) + Antenna gain (dBi) – Cable loss (dB), as shown in Figure 12-8.

EIRP is important from a resulting power and regulations standpoint. Most countries allow a maximum Tx power of the transmitter and a final maximum EIRP value, which is the resulting power when the antenna is added. An installer must pick the appropriate antenna and transmitter power settings, based on regulations for the country of deployment.

For example, say that you want to calculate the EIRP for a deployment with the following parameters:

- Tx power = 10 dBm
- Antenna gain = 6 dBi
- Cable loss = –3 dB

The EIRP in this case is calculated as 10 + 6 – 3 = 13 dBm.

IEEE Wireless Standards

This section discusses the IEEE 802.11 wireless communication standards for channels, data rates, and transmission techniques in Wi-Fi devices.

802.11 Standards for Channels and Data Rates

Being able to use a band, or range, of frequencies does not mean you can use it in any way you like. Important elements, such as which modulation technique to use, how a frame should be coded, which type of headers should be in the frame, and what the physical transmission mechanism should be, must be defined in order for devices to communicate with one another effectively.

The IEEE 802.11 standard defines how Wi-Fi devices should transmit in the Industrial, Scientific, and Medical (ISM) band. Today, whenever a Wi-Fi device is used, its Layer 1 and Layer 2 functionalities—such as receiver sensitivity, MAC layer performance, data rates, and security—are defined by an IEEE 802.11 series protocol.

802.11b/g

The 802.11b standard, which was ratified in 1999, offers rates of 5.5 to 11 Mbps and operates in the 2.4 GHz spectrum.

The IEEE 802.11g standard, which was ratified in June 2003, operates in the same spectrum as 802.11b and is backward compatible with the 802.11b standard. 802.11g supports additional data rates of 6, 9, 12, 18, 24, 36, 48, and 54 Mbps. 802.11g delivers the same 54 Mbps maximum data rate as 802.11a but operates in the same 2.4 GHz band as 802.11b.

802.11b and 802.11g once had broad user acceptance and vendor support, but due to their use of the 2.4 GHz band, which is prone to interference from other devices, and the slower speeds than are available with the newer 802.11 standards, the 802.11b/g standards are rarely used in today's enterprise networks.

802.11a

The IEEE ratified the 802.11a standard in 1999. 802.11a, which delivers a maximum data rate of 54 Mbps, uses orthogonal frequency-division multiplexing (OFDM), which is a multicarrier system (as opposed to a single-carrier system). OFDM allows subchannels to overlap, so it provides high spectral efficiency, and the modulation technique that is allowed in OFDM is more efficient than the spread-spectrum techniques used with 802.11b. Operating in an unlicensed portion of the 5 GHz radio band, 802.11a is also immune to interference from devices that operate in the 2.4 GHz band.

Since this band is different from the 2.4 GHz-based products, 802.11a chips were initially expensive to produce. With 802.11g providing the same speed at 2.4 GHz and at longer distances, 802.11a has never had broad user acceptance. The slower speeds available with 802.11a than with the newer 802.11 standards has led to the 802.11a standard rarely being used in today's enterprise network.

802.11n

802.11n, which was ratified in September 2009, is backward compatible with 802.11a and 802.11b/g and provides throughput enhancements. Features include channel bonding for up to 40 MHz channels, packet aggregation, and block acknowledgment. In addition, improved signals thanks to multiple-input, multiple-output (MIMO) enable clients to connect with faster data rates at a given distance from the AP compared to 802.11a/b/g. The 802.11n MIMO antenna technology extends data rates into the hundreds of megabits per second in the 2.4 and 5 GHz bands, depending on the number of transmitters and receivers that the devices implement.

802.11ac

IEEE 802.11ac was ratified in December 2013. Like 802.11a, it operates in the 5 GHz spectrum. The initial deployment, "Wave 1," uses channel bonding for up to 80 MHz channels, 256-QAM coding, and one to three spatial streams, providing data rates up to 1.27 Gbps. "Wave 2" uses up to 160 MHz channel bonding, one to eight spatial streams, and multi-user MIMO (MU-MIMO), providing data rates up to 6.77 Gbps.

An 802.11ac device supports all mandatory modes of 802.11a and 802.11n. So, an 802.11ac AP can communicate with 802.11a and 802.11n clients using 802.11a- or 802.11n-formatted packets, and it is as if the AP were an 802.11n AP. Similarly, an 802.11ac client can communicate with an 802.11a or 802.11n AP by using 802.11a or 802.11n packets. Therefore, 802.11ac clients do not cause issues with an existing infrastructure.

802.11ax (Wi-Fi 6)

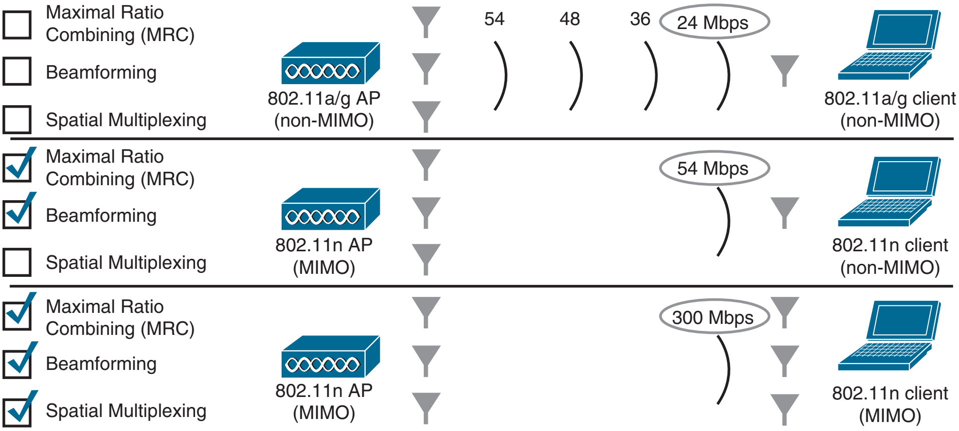
IEEE 802.11ax is a standards draft that is expected to be ratified in late 2020. The Wi-Fi Alliance has branded the standard Wi-Fi 6. The first wave of IEEE 802.11ax access points support eight spatial streams and 80 MHz channels, deliver up to 4.8 Gbps at the physical layer. Unlike 802.11ac, 802.11ax is a dual-band 2.4 and 5 GHz technology, so legacy 2.4 GHz–only clients can take advantage of its benefits. Wi-Fi 6 will also support 160 MHz–wide channels and will be able to achieve the same 4.8 Gbps speeds with fewer spatial streams.

Like 802.11ac, the 802.11ax standard also supports downlink MU-MIMO, which means a device can transmit to multiple receivers concurrently. However, 802.11ax also supports uplink MU-MIMO, which means a device can simultaneously receive from multiple transmitters.

802.11n/802.11ac MIMO

Today, APs and clients that support only the 802.11a/b/g protocols are considered legacy systems. Such a system uses a single transmitter that talks to a single receiver to provide a connection to the network. A legacy device that uses single-input, single-output (SISO) has only one radio that switches between antennas. When receiving a signal, the radio determines which antenna provides the strongest signal and switches to the best antenna. However, only one antenna is used at a time. This is illustrated at the top of Figure 12-9.

Figure 12-9 802.11n and 802.11ac MIMO


☐ Maximal Ratio Combining (MRC)
☐ Beamforming
☐ Spatial Multiplexing
802.11a/g AP (non-MIMO)
54
48
36
24 Mbps
802.11a/g client (non-MIMO)
☑ Maximal Ratio Combining (MRC)
☑ Beamforming
☐ Spatial Multiplexing
802.11n AP (MIMO)
54 Mbps
802.11n client (non-MIMO)
☑ Maximal Ratio Combining (MRC)
☑ Beamforming
☑ Spatial Multiplexing
802.11n AP (MIMO)
300 Mbps
802.11n client (MIMO)

This configuration leaves both the AP and the client susceptible to degraded performance when confronted by reflected copies of the signal—a phenomenon that is known as *multipath reception*.

802.11n/ac makes use of multiple antennas and radios, which are combined with advanced signal-processing methods to implement a technique known as multiple-input, multiple-output (MIMO). Several transmitter antennas send several frames over several paths. Several receiver antennas recombine these frames to optimize throughput and multipath resistance.

MIMO effectively improves the reliability of a Wi-Fi link, provides better SNR, and therefore reduces the likelihood that packets will be dropped or lost.

When MIMO is deployed only in APs, the technology delivers significant performance enhancements (as much as 30% over conventional 802.11a/b/g networks) even when communicating only with non-MIMO 802.11a/b/g clients by using a feature called Cisco ClientLink.

For example, at the distance from the AP at which an 802.11a or 802.11g client communicating with a conventional AP might drop from 54 to 24 Mbps, the same client communicating with a MIMO-enabled AP might be able to continue operating at 54 Mbps. This is illustrated in the middle of Figure 12-9.

Ultimately, 802.11 networks that incorporate both MIMO-enabled APs and MIMO-enabled Wi-Fi clients deliver dramatic gains in reliability and data throughput, as illustrated at the bottom of Figure 12-9.

MIMO incorporates three main technologies:

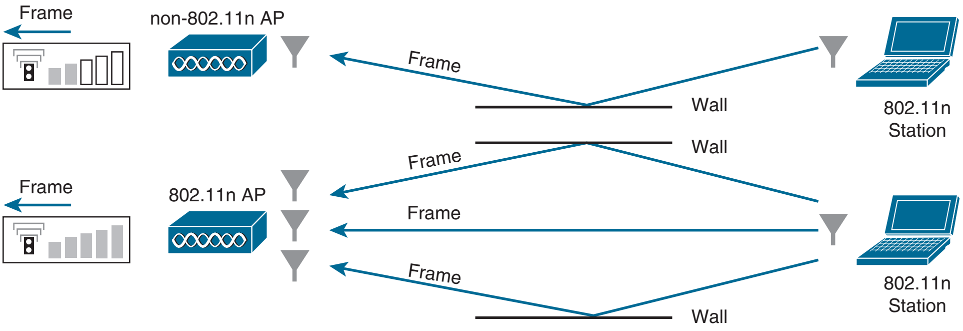
- Maximum-ratio combining (MRC)
- Beamforming
- Spatial multiplexing

Maximal Ratio Combining

A receiver with multiple antennas uses maximum-ratio combining (MRC) to optimally combine energies from multiple receive chains. An algorithm eliminates out-of-phase signal degradation.

Spatial multiplexing and Tx beamforming are used when there are multiple transmitters. MRC is the counterpart of Tx beamforming and takes place on the receiver side—usually on the AP—regardless of whether the client sender is 802.11n compatible. The receiver must have multiple antennas to use this feature, and 802.11n APs usually do. The MRC algorithm determines how to optimally combine the energy that is received at each antenna so that each signal that transmits to the AP circuit adds to the others in a coordinated fashion. In other words, the receiver analyzes the signals it receives from all its antennas and sends the signals into the transcoder so that they are in phase, thereby adding the strength of each signal to the other signals. At the top of Figure 12-10, only one weak signal is received by the AP. At the bottom of the figure, the AP receives three signals from the station. MRC combines these individual signals, allowing for faster data rates to be maintained between AP and client.

Figure 12-10 Maximum-Ratio Combining



Note that MRC is not related to multipath. Multipath issues are due to one antenna receiving reflected signals out of phase. This out-of-phase result, which is destructive to the signal quality, is transmitted to the AP. MRC combines the signals that come from two or three physically distinct antennas in a timely fashion so that the signals that are received on the antennas are in phase. The system evaluates the state of the channel for the signal that is received on each antenna and chooses the best received signal for each symbol, thereby ignoring pieces of waves on one chain that would not be read well. This system increases the quality of the reception. If you have, for example, three receive chains, you have three chances to read each symbol that is received, so MRC can minimize the possibility of interference degrading the section of the wave on all three receivers.

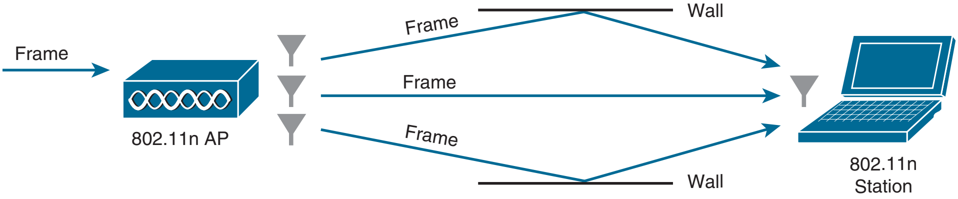
Multipath might still play a role. For example, each antenna might receive a reflected signal out of phase, and the antenna will be able to transmit to the AP only what it receives. The main advantage of MRC in this case is that, because each antenna is physically separated from the others, the received signals on each antenna are affected differently by multipath issues. When all the signals are added together, the result will be closer to the wave that was sent by the sender, and the relative impact of multipath on each antenna will be less predominant.

Beamforming

Tx beamforming is a technique that is used when there is more than one Tx antenna. The signals sent from the antennas can be coordinated so that the signal at the receiver is dramatically improved, even if the antenna is far from the sender.

This technique is generally used when the receiver has only one antenna and when the reflection sources are stable in space (such as with a receiver that is not moving fast and an indoor environment), as illustrated in Figure 12-11.

Figure 12-11 Beamforming



An 802.11n-capable transmitter may perform Tx beamforming. This technique allows the 802.11n-capable transmitter to adjust the phase of the signal that is transmitted on each antenna so that the reflected signals arrive in phase with one another at the receive (Rx) antenna. This technique can be applied even on a legacy client that has a single Rx antenna. Having multiple signals arrive in phase with one another effectively increases the Rx sensitivity of the single radio of a legacy client. This technique is software-defined beamforming.

802.11n added the opportunity for the receiver to help the beamforming transmitter do a better job of beamforming. This process, called *sounding*, enables the beamformer to precisely steer its transmitted energy toward the receiver. 802.11ac defines a single protocol for one 802.11ac device to sound other 802.11ac devices. The protocol that is selected loosely follows the 802.11n explicit compressed feedback protocol.

Explicit beamforming requires that the AP and the client have the same capabilities. The AP dynamically gathers information from the client in order to determine the best path. Implicit beamforming uses some information from the client at the initial association. Implicit beamforming improves the signal for older devices.

802.11n originally specified how MIMO technology can be used to improve SNR at the receiver by using Tx beamforming. However, both the AP and the client need to support this capability.

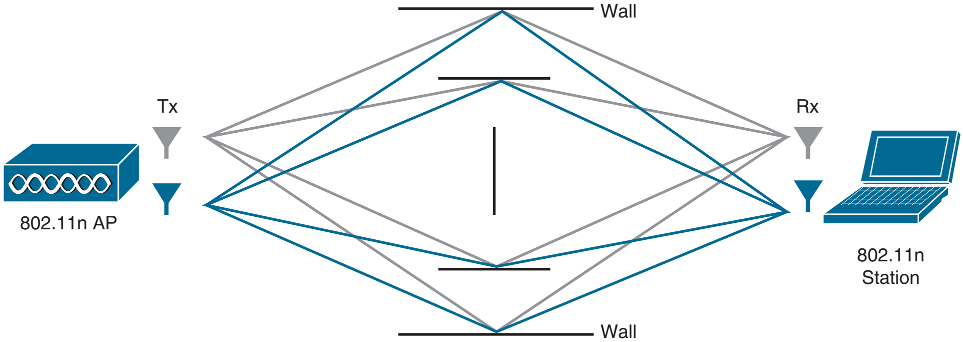
Cisco ClientLink technology helps solve the problems of mixed-client networks by making sure that older 802.11a/n clients operate at the best possible rates, especially when they are near cell boundaries. ClientLink also supports the ever-growing number of 802.11ac clients that support one, two, or three spatial streams. Unlike most 802.11ac APs, which improve only uplink performance, Cisco ClientLink improves performance on both the uplink and the downlink, providing better user experiences with web browsing, email, and file downloads. ClientLink technology is based on signal-processing enhancements to the AP chipset and does not require changes to network parameters.

Spatial Multiplexing

Spatial multiplexing requires both an 802.11n/ac-capable transmitter and an 802.11n/ac-capable receiver. It requires a minimum of two receivers and a single transmitter per band and supports as many as four transmitters and four receivers per band. Spatial multiplexing allows the advanced signaling processes of 802.11n to effectively use the reflected signals that are detrimental to legacy protocols. The reflected signals allow this technology to function. The reduction in lost packets improves link reliability, which results in fewer retransmissions. Ultimately, the result is more consistent throughput, which helps ensure predictable coverage throughout a facility.

With spatial multiplexing, a signal stream is broken into multiple individual streams, each transmitted from a different antenna, using its own transmitter. Because there is space between antennas, each signal follows a different path to the receiver. This phenomenon, known as *spatial diversity*, is illustrated in Figure 12-12. Each radio can send a different data stream from the other radios, and all radios can send at the same time, using a complex algorithm that is built on feedback from the receiver.

Figure 12-12 Spatial Multiplexing



The receiver has multiple antennas, each with its own radio. Each receiver radio independently decodes the arriving signals. Then each Rx signal is combined with the signals from the other radios. Thanks to a lot of complex math, the result is a much better Rx signal that can be achieved with either a single antenna or with Tx beamforming. Using multiple streams allows 802.11n devices to send redundant information for greater reliability, a greater volume of information for improved throughput, or a combination of the two.

For example, consider a sender that has two antennas. The data is broken into two streams that two transmitters transmit at the same frequency. The receiver says, "Using my three Rx antennas with my multipath and math skills, I can recognize the two streams that are transmitted at the same frequency because the transmitters have spatial separation."

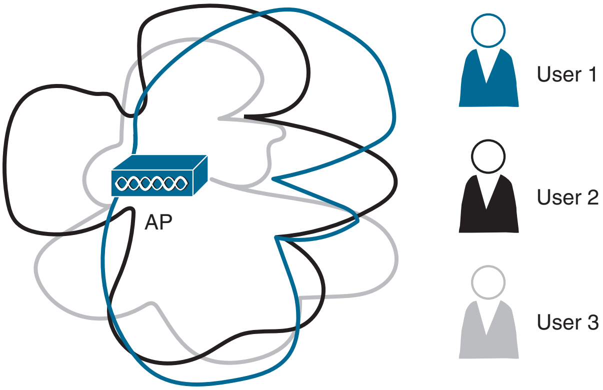
A Wi-Fi network is more efficient when it uses MIMO spatial multiplexing, but there can be a difference between the sender and the receiver. When a transmitter can emit over three antennas, it is described as having three data streams. When it can receive and combine signals from three antennas, it is described as having three receive chains. The combination of three data streams and three receive chains is commonly denoted as three by three (3X3). Similarly, there are 2X2, 4X4, and 8X8 devices, which have two, four, and eight spatial streams, respectively.

An 802.11ac environment allows more data by increasing the spatial streams to up to eight. Therefore, an 80 MHz channel with one stream provides a throughput of 300 Mbps, while eight streams provide a throughput of 2400 Mbps. Using a 160 MHz channel would allow throughput of 867 Mbps (one stream) to 6900 Mbps (eight streams).

802.11ac MU-MIMO

With 802.11n, a device can transmit multiple spatial streams at once—but only directed to a single address. For individually addressed frames, this means that only a single device (or user) receives data at a time. This is called *single-user MIMO* (*SU-MIMO*). 802.11ac provides for a feature called *multi-user MIMO* (*MU-MIMO*), in which an AP uses its antenna resources to transmit multiple frames to up to four different clients, all at the same time and over the same frequency spectrum, as illustrated in Figure 12-13.

Figure 12-13 MU-MIMO Using a Combination of Beamforming and Null Steering to Multiple Clients in Parallel



To send data to user 1, the AP forms a strong beam toward user 1 (shown as the top-right lobe of the color curve). At the same time, the AP minimizes the energy for user 1 in the direction of user 2 and user 3. This circumstance is called *null steering* and is shown as the color notches. In addition, the AP is sending data to user 2, forms a beam toward user 2, and forms notches toward users 1 and 3, as shown by the color curve. The color curve shows a similar beam toward user 3 and nulls toward users 1 and 2. In this way, each of users 1, 2, and 3 receives a strong copy of the desired data that is only slightly degraded by interference from data for the other users.

MU-MIMO allows an AP to deliver appreciably more data to its associated clients, especially for small form-factor clients (often BYOD clients) that are limited to a single antenna each. If the AP is transmitting to two or three clients, the effective speed increase varies from a factor of unity (no speed increase) up to a factor of two or three times, depending on the Wi-Fi channel conditions. If the speed-up factor drops below unity, the AP uses SU-MIMO instead.

Study Resources

For today's exam topics, refer to the following resource for more study.

Resource	**Module, Chapter, or Link**
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	17

Day 11

Wireless Deployment

ENCOR 350-401 Exam Topics

Architecture

- Analyze design principles of a WLAN deployment
    - Wireless deployment models (centralized, distributed, controller-less, controller based, cloud, remote branch)

Infrastructure

- Wireless
    - Describe AP modes and antenna types
    - Describe access point discovery and join process (discovery algorithms, WLC selection process)

Key Topics

Today we continue our review of wireless concepts by focusing on WLAN deployment architectures and looking at AP modes and antenna types that are commonly used in enterprise campus networks. Today we also discuss the process that a lightweight AP must go through to discover and bind to a wireless LAN controller.

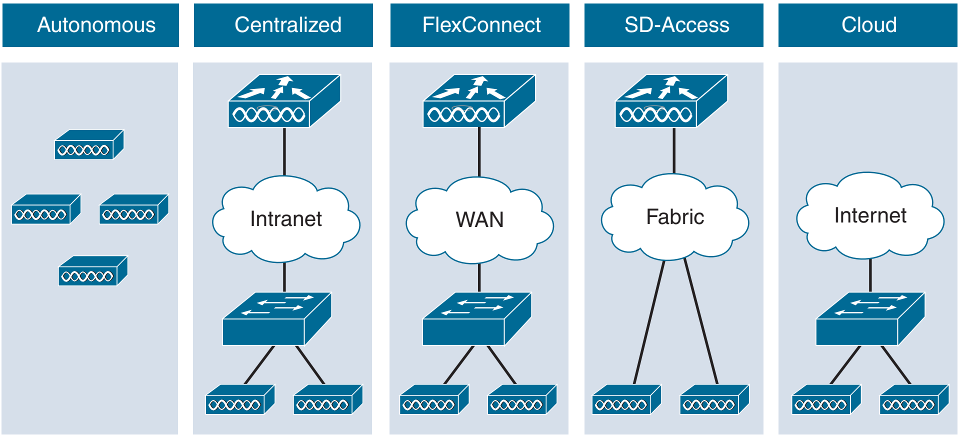
The number of wireless deployment choices can be daunting for a wireless network design engineer. Wireless LAN controllers (WLCs) have different capabilities and form factors, and there is the choice of on-premises or cloud-based management. Today we explore several wireless deployment options, looking at how they operate, where they are best used, and their benefits and limitations.

Today we also look at the Catalyst 9800 Series wireless controller deployment options, as well as the Cisco Mobility Express solution for 802.11ac Wave 2 APs.

Wireless Deployment Overview

There are several types of WLAN deployments, as illustrated in Figure 11-1:

Figure 11-1 Wireless Deployment Options



- **Autonomous:** This type of deployment is used for hotspots or smaller enterprises. Each AP is managed individually. This option is simple to deploy and cost-effective.
- **Centralized:** This type of deployment is used primarily in campus network environments where traffic is centralized. APs physically connect to switches, which connect to centralized WLCs.
- **Cisco FlexConnect:** This type of deployment is designed for enterprises that have branch or remote offices, in locations with a relatively small number of APs where deployment of WLCs is not justified or desired. WLAN data traffic is either tunneled back to a central WLC (central switching), or the data traffic is broken out locally at the wired interface (local switching) of the AP.
- **SD-Access:** This type of deployment allows for the integration of wireless access in the SD-Access architecture. This option moves the enterprise network from using a VLAN-centric architecture to using a user group–based enterprise architecture, with flexible Layer 2 extensions within and across sites. This solution offers automated network provisioning via the Cisco DNA Center application. SD-Access wireless deployment options are discussed on Day 18, "SD-Access."
- **Cloud-managed:** Cloud-based virtual controllers are offered via the Cisco Meraki or the Cisco 9800 cloud controller solutions. This type of deployment offers centralized installation and management, scales from small branches to large networks, and reduces operational costs.

The following sections look at each of these options in more detail.

Autonomous AP Deployment

In an autonomous AP deployment, both the WLAN and AP operate as and are managed as independent entities in the network. Because each AP operates independently from, and has no knowledge of, any other AP(s) in the network, each one can be thought of as an independent cell. This type of wireless infrastructure is a network of convenience and is not considered business critical.

The greatest benefit of an autonomous AP deployment is that it is a simple and cost-effective way to extend an existing wired infrastructure for a small network. If a customer plans to implement a Wi-Fi network in a small office, a store, or another small location and wants to provide Wi-Fi coverage in a few meeting rooms, a warehouse, and a lobby, the Wi-Fi network designer may consider deployment of a small number of autonomous APs installed on premises after conducting a site survey.

Some customers prefer autonomous deployments to meet their needs. These customers typically either need to provide connectivity in a small area or have low-bandwidth requirements and need to support only limited-function devices, such as hand scanners or simple monitoring equipment. In such deployments, clients generally connect to APs that are directly connected to switches on the LAN, in essence converting the wired network to wireless.

Although autonomous APs were originally the only choice for Wi-Fi network design and implementation, today they represent a smaller percentage of new installations.

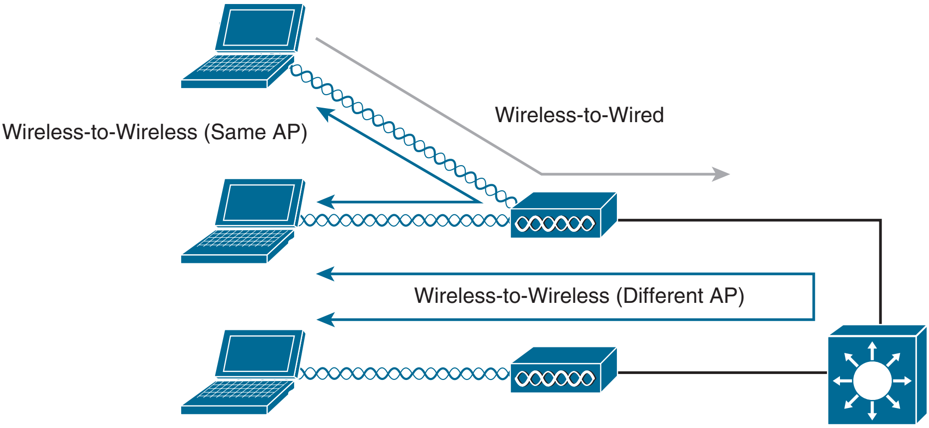
Because autonomous AP environments are so small, factors such as roaming and tightly synchronized AP transmit levels do not apply. Typical configuration parameters include the following:

- Service set identifier (SSID)
- Wireless security choice
- Transmit power levels to set the transmit power levels of the APs so that the signal does not propagate into adjacent building spaces belonging to other tenants or into the parking lot.

Autonomous Deployment Traffic Flow

Figure 11-2 illustrates three different traffic flow scenarios in an autonomous AP deployment:

Figure 11-2 Autonomous AP Traffic Flow


Wireless-to-Wired
Wireless-to-Wireless (Same AP)
Wireless-to-Wireless (Different AP)

- **Wireless-to-wired:** Wireless-to-wired client traffic transits directly through the AP. Client traffic flows across the wireless interface, through the access point, converting an 802.11 frame into an 802.3 frame and sending the frame to the local access switch to which the access point is connected. Because of this type of deployment, most early implementations of wireless networks were merely considered extensions of wired networks.

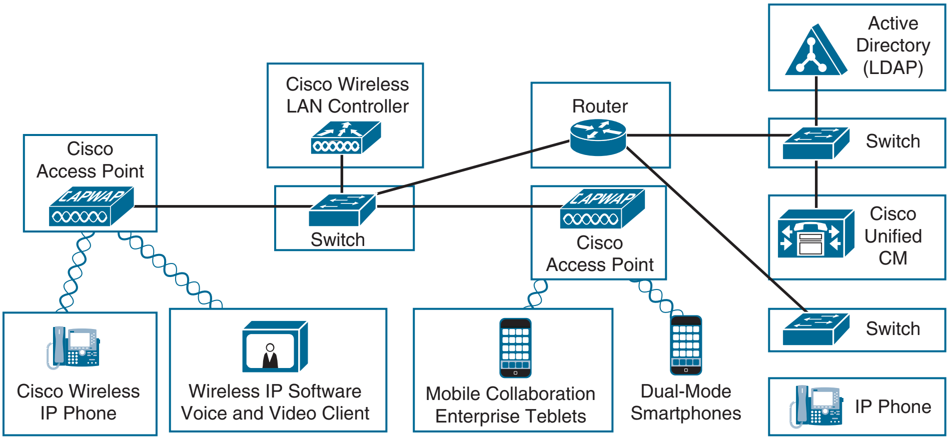
- **Wireless-to-wireless (same AP):** Traffic from a wireless client to another wireless client connected to the same AP (through the same WLAN or, sometimes, different WLANs, and even different radios) in the same subnet goes from one client to the other client through the AP, without going beyond the AP to the switch. This type of traffic does not create any load on the switch supporting the AP. If two clients are in different VLANs (and therefore are in different WLANs on the same radio or on different radios with different WLAN-to-VLAN mappings), the AP cannot route clients between VLANs. The AP must forward the traffic coming from the first client to the LAN until it reaches a router that forwards the packet to the second client subnet. The frame then transits back to the AP, to be delivered to the second client. In this configuration, each client communicates with the other client as if the other client were on the LAN, behind a router. Clients do not see each other in the same wireless space.
- **Wireless-to-wireless (different APs):** Traffic from clients sending frames to other clients connected on other APs must transit through the wired infrastructure in most cases. Although the AP-to-AP wireless communication is a technical possibility, this solution offers less efficiency than transiting through a wire and is reserved for specific cases (building-to-building wireless links, for example). In all other cases, even if both APs attach to the same switch in the same subnet and serve the same WLAN for both clients, traffic transits through the switch, and each client has no awareness that the other client is in the same WLAN.

This data flow type has an important impact on network design. Wireless device-to-wireless device communication may heavily involve the wired infrastructure if the clients are not in the same WLAN on the same AP. A router is needed if a network is designed to have clients in different VLANs. On the other hand, with too many clients in the same VLAN, congestion issues may arise due to clients receiving too many broadcasts. Designing smaller VLANs is usually recommended, but it is important to be mindful of the impact of such a design on the wired network load.

Centralized Cisco WLC Deployment

The unified, or centralized, architecture involves a Cisco WLC that is responsible for configuration, control, and management of several APs to which client devices connect. Figure 11-3 shows a simple centralized WLC deployment.

Figure 11-3 Centralized WLC Deployment Model


Cisco Wireless LAN Controller
Router
Active Directory (LDAP)
Cisco Access Point
CAPWAP
Switch
CAPWAP
Cisco Access Point
Switch
Cisco Unified CM
Switch
Cisco Wireless IP Phone
Wireless IP Software Voice and Video Client
Mobile Collaboration Enterprise Teblets
Dual-Mode Smartphones
IP Phone

Unlike autonomous APs, the APs in a centralized architecture do not function independently. They have reduced functionality in the AP and depend on the WLC to configure, control, and manage several APs. In this architecture, the Wi-Fi functionality is split between the AP and the WLC, which resides deeper in the LAN architecture. The APs handle only real-time MAC functionality, and all non-real-time MAC functionality is processed by the WLC.

The primary advantage of such a centralized architecture is that it provides network administrators with a structured and hierarchical mode of control for multiple APs in the network. This Wi-Fi architecture eases management of large deployments in enterprise networks. APs have visibility and awareness of the neighboring APs and allow the use of additional functionalities.

A WLC can be informed if one of the APs becomes faulty and neighboring APs adjust power levels to compensate for the faulty AP. Also, a Cisco WLC can offload Wi-Fi clients to a neighboring AP if one of the APs becomes overloaded. Load balancing and self-healing are just two of the important applications of centralized architecture.

The Wi-Fi network design for a centralized deployment depends on customer Wi-Fi requirements. The centralized architecture would be a good choice for a customer that is planning to expand the existing Wi-Fi network and provide coverage for an entire building with several floors. This scenario might also include the deployment of many applications that would require centralized management.

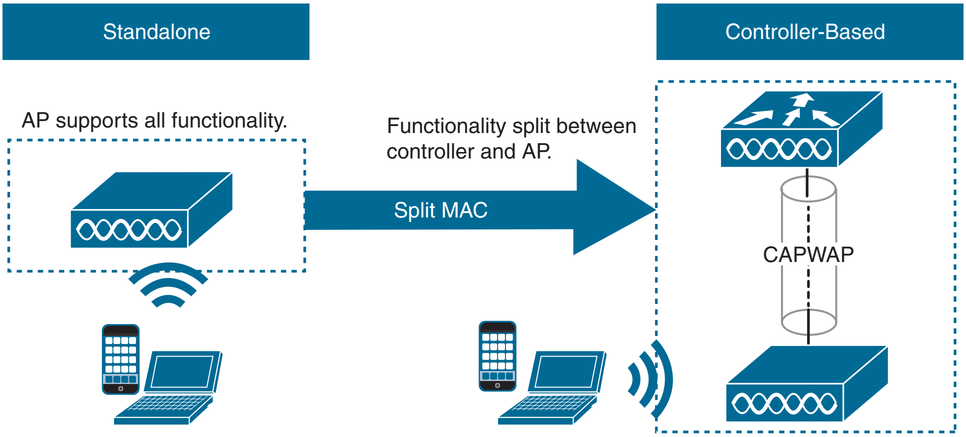
There are certain limitations to the centralized deployment architecture. All end-user traffic is forwarded to the WLC, which means the WLC can become a bottleneck and a single point of failure.

Note that these limitations are usually not considered severe enough to prefer autonomous solutions to centralized solutions in medium to large deployments. A design can typically compensate for such limitations. You can design your network structure to avoid bottleneck and congestion issues at the controller level. You can deploy efficient links between the APs and the controller for networks where client data latency is an issue. In such cases, the controllers are usually close to the APs, separated from them by only one or a few fast LAN switches or an efficient WAN (for example, with MPLS). You can configure the APs to fall back to a secondary controller of your choice when the primary controller is down. Consistent configuration between controllers ensures that no client data service is delayed or interrupted.

Split MAC

In an enterprise with numerous APs, a controller-based architecture provides advanced WLAN features and benefits that are easy to deploy, scale, and manage. The controller-based architecture allows the splitting of 802.11 functions between the AP, which processes real-time portions of the protocol, and the Cisco WLC, which manages items that are not time-sensitive. This model, called *split MAC*, is illustrated in Figure 11-4.

Figure 11-4 Split MAC Architecture


Standalone
Controller-Based
AP supports all functionality.
Functionality split between controller and AP.
Split MAC
CAPWAP

The AP processes the portions of the protocol that have real-time requirements, including the following:

- Frame-exchange handshaking between client and AP when transferring a frame.
- Transmission of beacon frames. Beacons are essentially the heartbeats of a wireless network. The AP of a wireless network broadcasts beacons, and clients listen for the beacon frames.
- Buffering and transmission of frames for clients in a power-save operation.
- Responding to probe request frames from clients and forwarding notifications of received probe requests to the WLC. A probe request is used when the client is actively scanning to find APs.
- Providing real-time signal quality information to a Cisco WLC with every received frame.
- Monitoring all radio channels for noise, interference, and other WLANs and monitoring for the presence of other APs.
- Providing wireless encryption and decryption of 802.11 frames.

All remaining functionality is managed in the Cisco WLC, where time sensitivity is not a concern and WLC-wide visibility is required. The Cisco WLC provides the following functions for a wireless network:

- Authentication (PSK, EAP, and so on)
- Radio frequency (RF) management (that is, control of RF space for APs and clients)
- Client IP addressing
- Seamless roaming (Layer 2 or Layer 3 client roaming)
- QoS for voice, video, and data

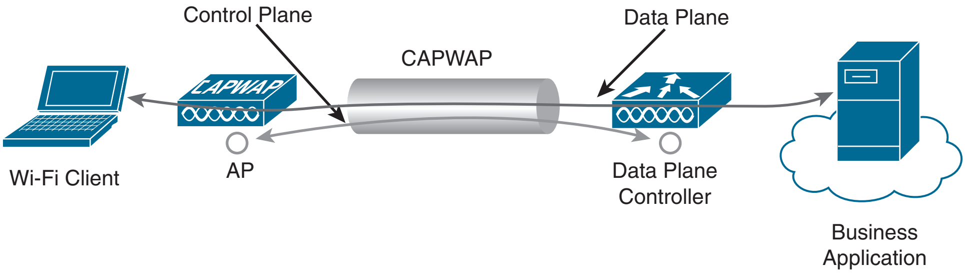
- AP configuration management (of VLANs, IP addresses, and so on)
- AP image management (current and consistent software levels across the enterprise)

CAPWAP

Control and Provisioning of Wireless Access Points (CAPWAP) is an open protocol that enables a WLC to manage a collection of wireless APs. CAPWAP control messages are exchanged between the WLC and the AP across an encrypted tunnel. This protocol includes the WLC discovery and joins process, AP configuration and firmware push from the WLC, and statistics gathering and wireless security enforcement. CAPWAP-capable APs must be authenticated before they are allowed to download any configuration from a WLC.

CAPWAP differentiates between the control plane and the data plane, as illustrated in Figure 11-5. The AP and WLC build a secure DTLS tunnel for control plane communication. WLC control plane messages are used to support wireless station access, authentication, and mobility.

Figure 11-5 Control and Data Plane CAPWAP Communication



Client data is encapsulated with a CAPWAP header that contains valuable information about the client RSSI and SNR and then is sent to the WLC, which forwards the data as needed. The data plane is not encrypted by default, but encryption can be added by using DTLS.

Centralized Deployment Traffic Flow

Wireless client data flow in a centralized architecture is a little bit more complex than in an autonomous architecture because the situation depends on how you configure your access points and your inter-controller communication.

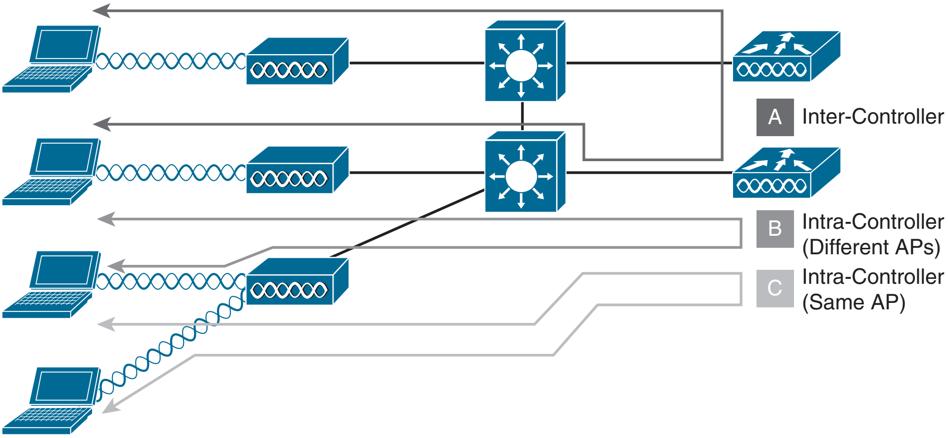
In a standard model where access points are in local mode, all wireless client data traffic is first sent to the controller, and the controller then decides on the policy (for example, ACL, QoS) to apply to the incoming traffic before deciding how to forward the client traffic to its final destination.

When the client wireless frame reaches the AP, the frame is encapsulated in CAPWAP and then forwarded to the controller. This encapsulation, which occurs for any WLAN supported by the AP, means that between the AP and the controller, CAPWAP data packets are sent from the AP's IP address, with a random client port, to the controller AP Manager IP address, to CAPWAP data port UDP 5247. The same logic applies to return traffic: It is sent from the controller AP Manager IP address (or the management interface, if the controller does not use AP manager interfaces) with a random client port, to the AP's IP address on CAPWAP port UDP 5247. The details of the client VLAN and WLAN are hidden inside the encapsulated payload. You also see CAPWAP control traffic between the AP and the controller that uses the CAPWAP control destination port UDP 5246.

When the encapsulated frame coming from the AP reaches the controller, the controller removes the CAPWAP encapsulation, decides on the policy to apply to this traffic (for example, QoS, ACL), and forwards the packet to the controller physical port associated with the VLAN for this client. This controller VLAN interface may also have policies (ACLs) that are applied at this point. Traffic leaves the controller from the port associated with the VLAN interface.

Figure 11-6 illustrates three different traffic flow scenarios in a centralized deployment.

Figure 11-6 Centralized Deployment Traffic Flow


A Inter-Controller
B Intra-Controller (Different APs)
C Intra-Controller (Same AP)

- **Inter-controller:** In the example at the top of Figure 11-6, when the destination subnet is the VLAN associated with a WLAN on another controller, the controller does not route between VLANs and subnets, so a router uses the routing table to decide how best to reach that destination network. On the last hop to the subnet, the last-hop router sends an ARP request to resolve the MAC address of the destination IP address. The second controller acts as an ARP proxy and answers the request in the name of the wireless client. The last-hop router then forwards the frame to the second controller. The second controller converts the 802.3 frame into an 802.11 frame, encapsulates the frame into CAPWAP, and sends this frame from the frame AP manager interface to the AP with which the destination client is associated.
- **Intra-controller between APs:** In the example in the middle of Figure 11-6, the flow is a bit simpler as both clients are associated with different APs connected to the same controller. Each controller keeps a list of the clients associated with each AP managed by the controller. When the controller receives a CAPWAP-encapsulated data frame from an access point, after deciding on an initial QoS and security policy, the controller examines both the destination MAC address and destination IP address. If the destination MAC address is a wireless client of the controller in the same subnet as the first client, the 802.11 frame is re-encapsulated into CAPWAP and forwarded to the AP with which that client is associated. If the destination

MAC address is not a wireless client of the controller, or if the destination MAC address is another wireless client of the controller but on a different VLAN, the 802.11 frame is converted to 802.3 and forwarded to the controller interface associated with the VLAN in which the source wireless client resides.

This scheme means that the frame is always forwarded to the controller first (when APs are in local mode) but never reaches the switch behind the controller if the destination is a wireless client of the controller. This logic is true regardless of which AP is associated with the destination client.

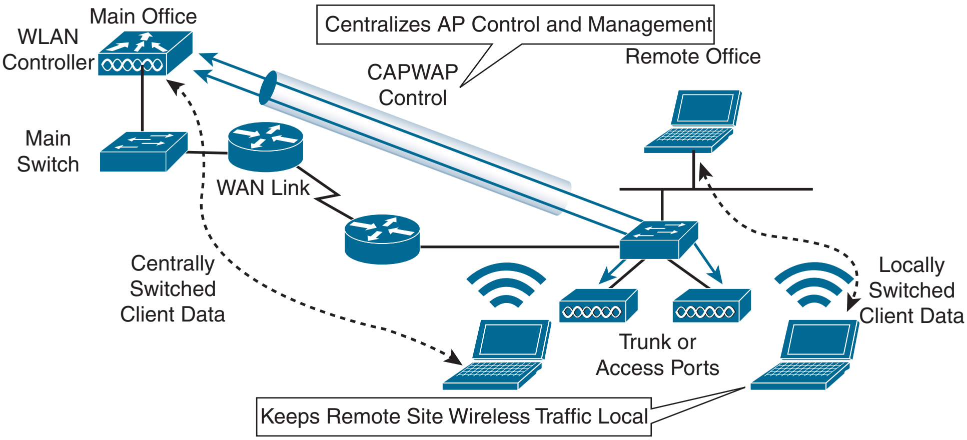
- **Intra-controller within the same AP**: In the example at the bottom of Figure 11-6, even if both clients are on the same AP, traffic is first forwarded to the controller before being sent back to the AP to be distributed to the client. This concept may be foreign to a customer, who might think that stopping at the AP and simply forwarding the frame back to the client would be more logical. But that approach does not account for the control given to the controller in a unified solution. The frame first must reach the controller so that QoS and security policies can be applied, along with bandwidth limitations, when necessary. This scheme is also the only way for an administrator to have a live view of wireless client traffic and conditions.

FlexConnect Deployment

The Cisco FlexConnect architecture is an extension of the centralized architecture designed for branch office and remote office deployments.

The FlexConnect architecture is best suited for locations with relatively small numbers of APs where deployment of a Cisco WLC is not justified or desired. It enables you to configure and control APs in a branch or remote office from the corporate office through a WAN link without the deployment of a controller in each office, as illustrated in Figure 11-7.

Figure 11-7 FlexConnect Deployment Model


Main Office
WLAN Controller
Centralizes AP Control and Management
CAPWAP Control
Remote Office
Main Switch
WAN Link
Centrally Switched Client Data
Locally Switched Client Data
Trunk or Access Ports
Keeps Remote Site Wireless Traffic Local

Clients connecting to APs at remote locations can be authenticated locally and can have their data bridged to the local Ethernet segment; this process is known as *local switching*. Alternatively, clients can have their traffic tunneled over the WAN via CAPWAP to a Cisco WLC at a central site to be switched to the network; this process is known as *central switching*. An AP can be configured to make a decision between local and central switching based on the destination of the traffic. With this process, known as *split tunneling*, the AP makes a decision on the most efficient switching method rather than allowing the client traffic to be needlessly sent across the WAN link via CAPWAP just to be returned over the WAN link as Ethernet traffic.

As a flexible WLAN architecture, FlexConnect is a good choice for small to medium-size branch offices with up to 50 APs per site. When customers expand their business infrastructure and introduce remote offices that are connected to the corporate office via WAN links, FlexConnect architecture may also be a good option.

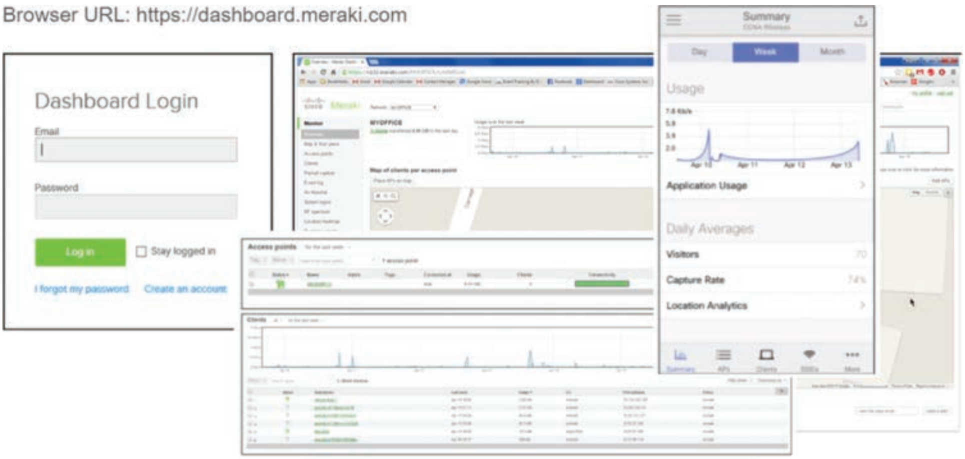
Cloud-Managed Meraki Deployment

The function of the wireless controller is no longer limited to an appliance in the network. The functionality can be located via software in a public cloud. The Cisco Meraki product portfolio provides a cloud-managed approach to IT that gives you browser access to an Internet, cloud-based controller for off-premises management of APs.

The Cisco Meraki deployment option offers wireless networking as a service for clients who require ease of use and management. The Cisco Meraki solution is well suited for organizations that are highly distributed, have lean IT teams, and are looking for simplicity.

The Cisco Meraki cloud architecture provides a feature set that is rich enough for large enterprise deployments yet easy enough to be used by IT generalists. Cloud-based infrastructure provides cost advantages in both small and large networks due to its ability to scale seamlessly. Figure 11-8 shows the Meraki dashboard GUI that is used to manage this cloud-based solution.

Figure 11-8 Cisco Meraki Dashboard



The Cisco Meraki solution is suited for the following use cases:

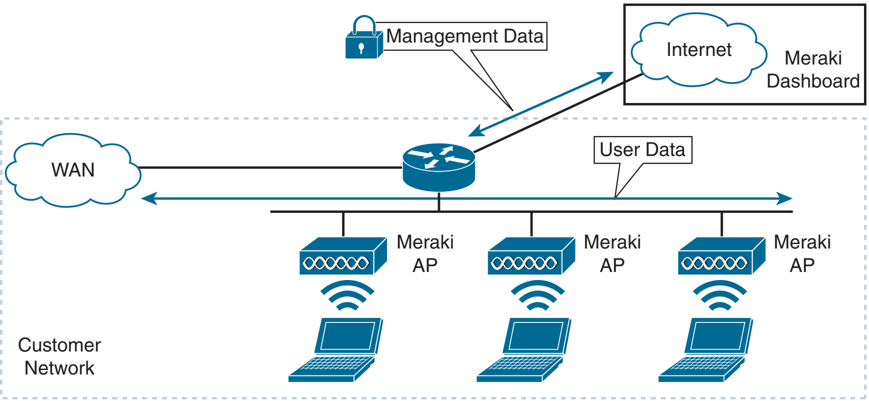
- Mid-market businesses and distributed sites
- Remote branches without on-site IT
- Organizations that may already be using cloud services

Cisco Meraki Deployment Traffic Flow

Cisco Meraki devices automatically connect to the Cisco Meraki cloud over a secure link, register with their network, and download their configuration. There is no need to manually stage APs or log in to switches for manual configuration and provisioning. Cisco Meraki APs use Auto RF to self-configure and optimize RF settings for maximum performance, even in dense and challenging environments.

Cisco Meraki uses an out-of-band management architecture, which means only management data flows through the Meraki cloud infrastructure. No user traffic passes through the Meraki data centers, and data stays on the private network, as illustrated in Figure 11-9.

Figure 11-9 Cisco Meraki Traffic Patterns



The customer network continues to function normally even if it cannot access the cloud. Users can authenticate, firewall rules remain in place, and traffic flows at full line rate. Only management functions (reports, configuration tools, and so on) are interrupted when the connection is broken.

Meraki provides firmware updates via seamless, over-the-web upgrades. Each quarter Cisco Meraki releases updates that contain new features, performance improvements, and other enhancements. Firmware upgrades are delivered securely over the web, and it is possible to reschedule or cancel these upgrades at any time.

Cisco Catalyst 9800 Series Controller Deployment Options

Cisco Catalyst 9800 Series wireless controllers are next-generation wireless controllers built for intent-based networking. Catalyst 9800 Series controllers are Cisco IOS XE based and integrate the RF excellence capabilities of Cisco Aironet with the intent-based networking capabilities of Cisco IOS XE.

Catalyst 9800 wireless controllers offer features such as the following:

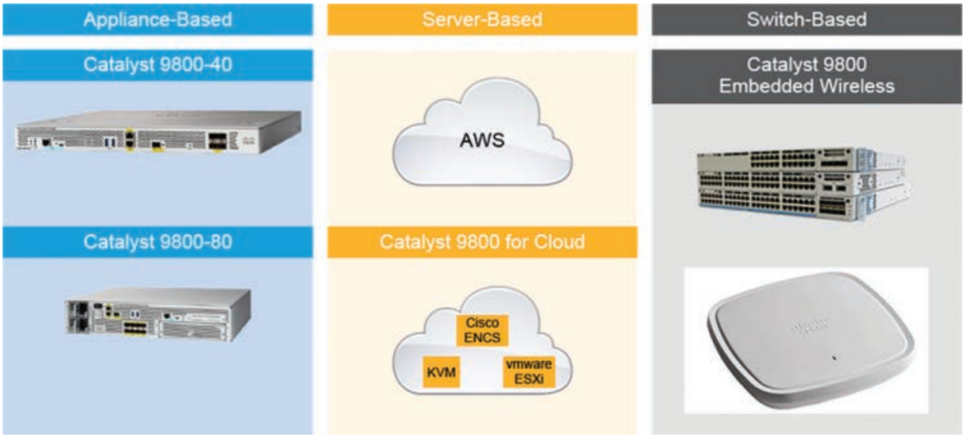
- The controllers come with high availability and seamless software updates that are enabled by hot and cold patching. This keeps your clients and services always on, during both planned and unplanned events.
- The controllers come with built-in security: secure boot, runtime defenses, image signing, integrity verification, and hardware authenticity.
- The controllers can be deployed anywhere to enable wireless connectivity, such as on an on-premise device, on cloud (public or private), or embedded on a switch.
- The controller can be managed using Cisco DNA Center, NETCONF/YANG, a web-based GUI, or the CLI.
- The controllers run a modular operating system. Open and programmable APIs enable the automation of your Day 0…*N* network operations. Model-driven streaming telemetry provides deep insights into your network and client health.

Catalyst 9800 Series wireless controllers are available in multiple form factors to cater to several deployment options, as illustrated in Figure 11-10:

- Catalyst 9800 Series wireless controller appliance
- Catalyst 9800 Series wireless controller for public or private cloud
- Catalyst 9800 embedded wireless controller for a switch
- Catalyst 9800 embedded wireless controller for an AP

The Catalyst 9800 Series wireless controller appliance deployment falls under the centralized deployment option discussed earlier.

Figure 11-10 Cisco Catalyst 9800 Series Wireless Controller Deployment Options

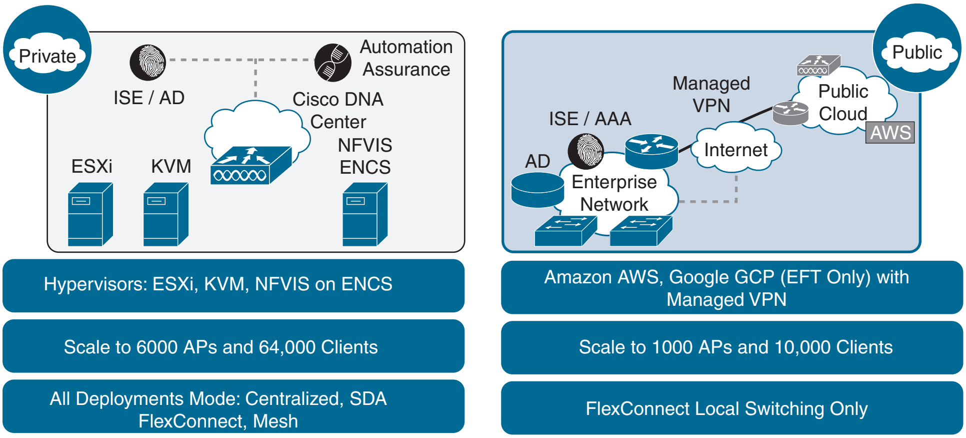


Catalyst 9800 Wireless Controller for Cloud

The Cisco Catalyst 9800-CL wireless controller for cloud is a virtual wireless controller that can be deployed on a Cisco Unified Computing System (UCS) server as a virtual machine (VM) instance on a Linux-based 64-bit guest operating system.

The virtual form factor of the Catalyst 9800 wireless controller can be deployed in a private cloud that supports ESXi, KVM, and NFVIS on ENCS hypervisors or in the public AWS cloud as infrastructure as a service (IaaS). Figure 11-11 compares some of the features of Catalyst 9800 wireless controllers for public and private clouds.

Figure 11-11 Cisco Catalyst 9800 Series Wireless Controller for Cloud: Private and Public Deployment Options


Private
ISE / AD
Automation
Assurance
Cisco DNA
Center
NFVIS
ENCS
ESXi
KVM
Hypervisors: ESXi, KVM, NFVIS on ENCS
Scale to 6000 APs and 64,000 Clients
All Deployments Mode: Centralized, SDA
FlexConnect, Mesh
Public
Public
Cloud
AWS
Managed
VPN
ISE / AAA
AD
Internet
Enterprise
Network
Amazon AWS, Google GCP (EFT Only) with
Managed VPN
Scale to 1000 APs and 10,000 Clients
FlexConnect Local Switching Only

The Catalyst 9800 wireless controller for cloud supports a subset of Cisco IOS XE Software features and technologies, providing Cisco IOS XE features on a virtualization platform. When a Catalyst 9800 controller is deployed as a VM, the Cisco IOS XE Software functions as if it were deployed on a traditional Cisco controller appliance.

A Catalyst 9800 wireless controller takes advantage of the benefits of virtualization to provide the following:

- **Hardware independence:** Because the controller runs on a VM, it can be supported on the x86 hardware that the virtualization platform supports.
- **Sharing of resources:** The resources used by the controller are managed by the hypervisor; these resources can be shared among VMs. The hardware resources that the VM server allocates to a specific VM can be reallocated to another VM on the server.
- **Flexibility in deployment:** You can easily move a VM from one server to another. Thus, you can move a controller from a server in one physical location to a server in another physical location without moving any hardware resources.

Catalyst 9800 Embedded Wireless Controller

There are now two Catalyst 9800 embedded wireless controller options available: embedded inside a Cisco Catalyst 9300 Series switch or embedded inside a Cisco Catalyst 9100 Series access point.

Catalyst 9800 Embedded Wireless for a Switch

Catalyst 9800 wireless controller software for the Cisco Catalyst 9300 switch brings the wired and wireless infrastructure together with consistent policy and management.

The Cisco Catalyst 9800 embedded wireless controller software package can be installed on Cisco Catalyst 9300 Series switches to enable wireless controller functionality for distributed branches and small campuses.

Once installed, the Catalyst 9800 embedded wireless controller running on a Catalyst 9300 Series switch can support up to 200 APs and 4000 clients. A maximum of 2 wireless controllers can be enabled per site on 2 different Catalyst 9300 Series switches, increasing the scale up to 400 APs and 8000 wireless clients.

The Catalyst 9800 embedded wireless controller software package has two supported topologies for Cisco SD-Access, as illustrated in Figure 11-12:

- **Option 1:** Catalyst 9300 Series switches functioning as co-located border and control plane devices.
- **Option 2:** Catalyst 9300 Series switches functioning as fabric in a box.

The embedded wireless controller deployment model supports only SD-Access, which is a highly secure solution for small campuses and distributed branches. The embedded controller supports access points only in fabric mode.

Catalyst 9800 Embedded Wireless for an AP

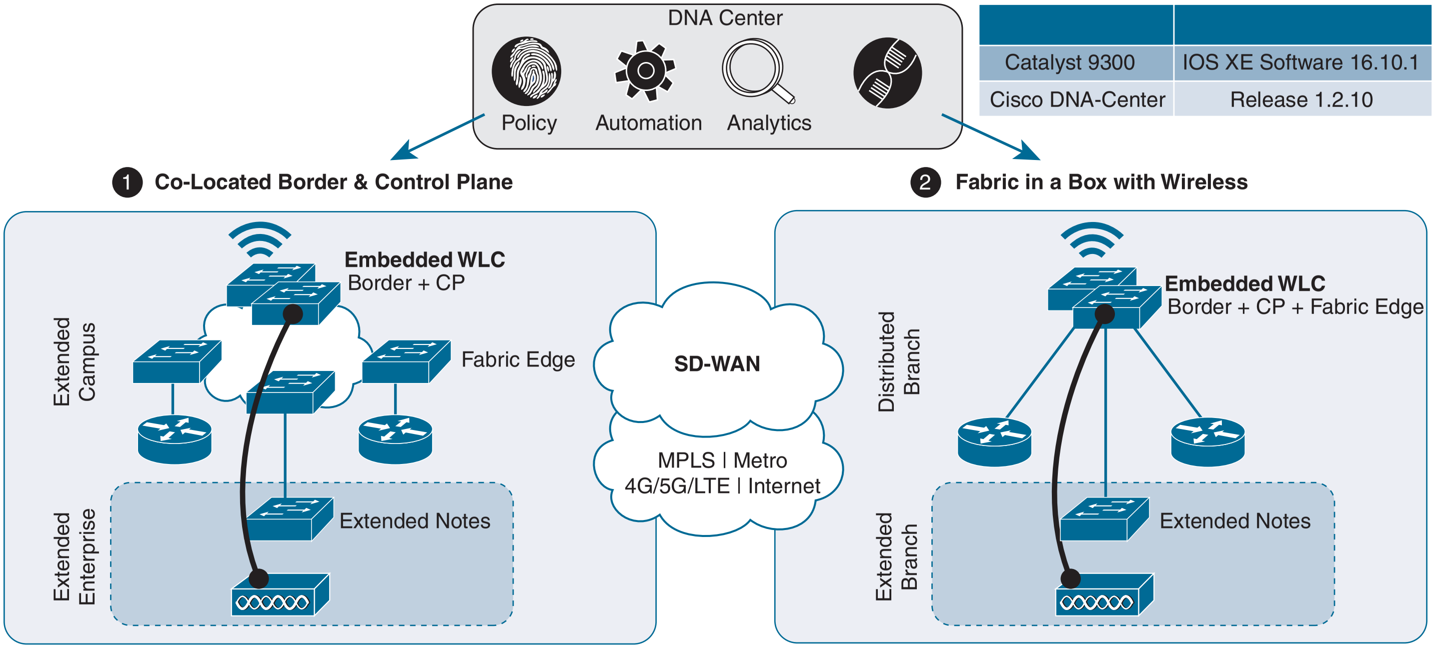
The Cisco Embedded Wireless Controller on Catalyst Access Point (EWC-AP) is a next-generation Wi-Fi solution that combines the Cisco Catalyst 9800 Series wireless controllers with the latest Wi-Fi 6 Cisco Catalyst 9100 access points.

Built for intent-based networking and Cisco DNA, the EWC-AP helps you reduce complexity, optimize IT, and reduce operational costs by leveraging intelligence, automation, and human expertise. Through Cisco DNA Center, analytics and assurance are made available on the EWC-AP though packet captures and RF analysis.

The latest EWC-AP (Catalyst 9130) can manage up to 100 APs and 2000 clients, for a maximum of 16 WLANs. It is best suited for distributed branches and small campuses that want enterprise-class features and want to upgrade to Wi-Fi 6 using minimal IT resources. It is possible to deploy 2 Catalyst 9100 Series EWC-APs in an active/standby pair to ensure resiliency for a wireless network. Furthermore, IOS XE supports hot and cold patching to avoid downtime during upgrades and migrations of either AP or controller functionalities.

Onboarding and management of EWC-AP can be handled through a local web user interface or through a smartphone mobile application. In addition, you can take advantage of programmability through REST APIs, NETCONF, and YANG. For larger deployments, the EWC-APs can also be managed via Cisco DNA Center.

Figure 11-12 Cisco Catalyst 9800 Series Embedded Wireless Controller



Catalyst 9300	IOS XE Software 16.10.1
Cisco DNA-Center	Release 1.2.10

DNA Center
Policy
Automation
Analytics
1 Co-Located Border & Control Plane
2 Fabric in a Box with Wireless
Extended Campus
Embedded WLC
Border + CP
Fabric Edge
Extended Enterprise
Extended Notes
SD-WAN
MPLS | Metro
4G/5G/LTE | Internet
Distributed Branch
Embedded WLC
Border + CP + Fabric Edge
Extended Branch
Extended Notes

Cisco Mobility Express

Cisco Mobility Express is a reliable and affordable wireless solution for enterprise branches or small to medium-size businesses that want a managed AP solution without being required to buy, maintain, and manage a separate WLAN controller appliance.

Cisco Mobility Express is a virtual wireless LAN controller that is integrated into an access point. By default, all access points run the Cisco Aironet CAPWAP image. To acquire the wireless LAN controller functionality, the Cisco Mobility Express image needs to be installed on a Cisco 802.11ac Wave 2 access point such as a Cisco Aironet 1800 or 2800 Series AP.

A Cisco Mobility Express solution consists of the following components:

- **Master access point:** The Cisco Aironet access point that runs the virtual wireless LAN controller function is called the *master access point*. Depending on the model, the master access point can manage up to 100 access points and 2000 clients. In addition to running the virtual WLC function, the master can also service clients at the same time.
- **Subordinate access point:** Cisco Aironet access points that are managed by the master access point in a Mobility Express network are called *subordinate access points*. Subordinate access points only service clients and do not actively run the wireless LAN controller function. Subordinate access points can be, but are not required to be, 802.11ac Wave 2 access points.

Wireless AP Operation

When a controller-based access point joins a controller, the controller manages the access point (AP) configuration, firmware, control transactions, and data transactions. This section explores the processes an AP goes through to find and associate with a controller and establish an association with another controller if its primary controller is no longer available, as well as the different modes of operation the AP can run in when it becomes part of the network.

Wireless LAN Controller Discovery Process

Once an AP obtains an IP address either statically or through DHCP, the AP can learn the management IP addresses of multiple controllers by using the following methods:

- **Subnetwork broadcast:** A Cisco controller-based AP can send a CAPWAP controller discovery request to the local subnetwork. All Cisco WLCs in the local subnetwork that receive this packet respond with a discovery response. This feature can also be enabled as a Layer 3 discovery process when controllers are on different subnets from the access point; it involves using IP addresses and UDP packets rather the MAC addresses, as are used by Layer 2 discovery.
- **Locally stored IPv4 or IPv6 address:** If an AP was previously associated with a controller, which can be set up through manual configuration, the IP addresses of the primary, secondary, and tertiary controllers are stored in the nonvolatile memory of the AP. Also, several controllers can be grouped to form a mobility group. Upon associating to a Cisco WLC, the AP learns the IP addresses of the other members of the mobility group. This information is stored in the AP and cached even after a reboot.

- **DHCP vendor option:** When a Cisco AP sends a DHCP discover request, it uses the Cisco vendor option, which is option 43 with IPv4 and option 52 with IPv6. If the DHCP server is configured to respond to these options, it sends a list of Cisco controller IP addresses to the AP in the DHCP ACK message. The AP then uses this information to send a unicast discovery request to each of the controllers. The vendor string that is used by APs to request the vendor option depends on the AP model.
- **DNS:** If you have configured your DHCP server to provide both option 006 (DNS server address) and option 015 (domain name) information, the AP can obtain WLC addresses from the DNS server. The process works as follows:
    1. The AP gets its IP address from DHCP with options 6 and 15 configured.
    2. The AP can obtain the IP address of the DNS server from the DHCP option.
    3. The AP uses this information to perform a hostname lookup using CISCO-CAPWAP-CONTROLLER.<*localdomain*>, which resolves to available WLC management interface IP addresses (IPv4 or IPv6, or both).
    4. The AP can then perform a directed message to associate to responsive WLCs.

To prevent all APs from joining a single controller based on a DNS name resolution, the domain name may vary; this is what is done to dispatch APs to different controllers across the enterprise network, based on different domain names that are configured in their respective DNS scopes.

AP Join Order

Once an AP has gathered a list of WLCs from the CAPWAP discovery process, it selects and joins one WLC via the primary, secondary, tertiary, or master configuration of controllers.

There is a predefined method by which an AP selects the controller with which to register:

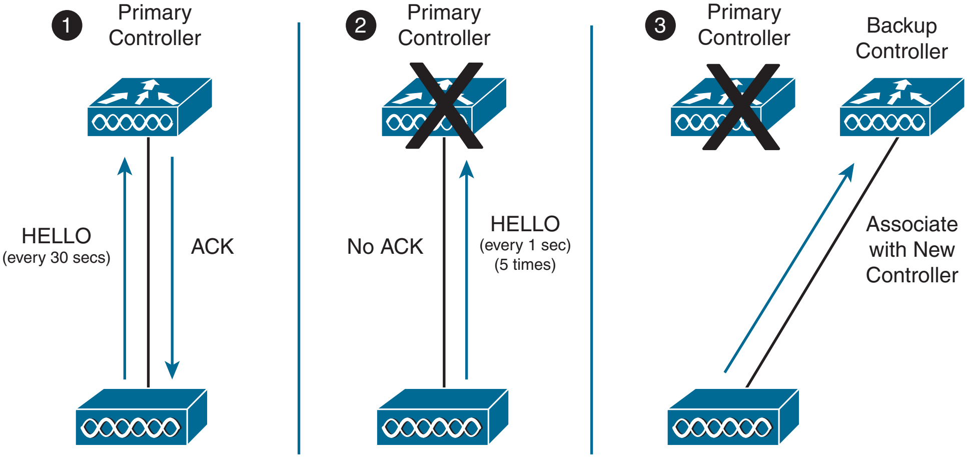
1. The AP associates first with its primary controller, assuming that it has been established.
2. Upon failing with the primary, it tries to associate with its secondary controller.
3. Upon failing with the secondary, it tries to associate with its tertiary controller.
4. If there is no controller information established in the AP, the AP looks for a master controller. The master controller option on the WLC is typically used to initialize a new AP for later deployment.
5. Finally, if there is no established controller and no master controller, the AP selects the least-loaded controller, which is defined as the controller with the greatest available AP capacity.

AP Failover

A WLC is designed to provide high availability for APs. In a WLC failure, the APs that are associated with the WLC migrate to other controllers that have enough capacity. The APs fall back to their primary controller when it is back online, assuming that AP fallback has not been disabled.

The AP uses a *hello packet*, also known as the *heartbeat*, to communicate with the controller and verify its reachability status. The default interval for the heartbeat is 30 seconds. Whenever one heartbeat ACK from the controller is missed, the AP resends the heartbeat up to five times at 1-second intervals. If an ACK is not received after the fifth retry, the AP declares the controller unreachable and searches for a new controller. This process is illustrated in Figure 11-13.

Figure 11-13 AP Failover



In addition to using this heartbeat process for failover, an AP maintains a list of backup controllers and periodically sends a primary discovery request to each entry on the list. Hellos are sent to these backup controllers every 120 seconds by default.

Failover Priority

Each controller has a defined number of communication ports for APs. When multiple controllers with unused AP ports are deployed on the same network and one controller fails, the dropped APs automatically poll for unused controller ports and associate with them.

You can configure a wireless network so that the backup controller recognizes a join request from a higher-priority AP and, if necessary, disassociates a lower-priority AP as a means to provide an available port.

By default, all APs are set to priority level 1, which is the lowest priority level. Therefore, you need to assign a priority level only to those APs that warrant a higher priority. To utilize this feature, you must enable failover priority on your network and assign priorities to the individual APs.

AP Fallback

When the WLC AP Fallback option is enabled, APs return to their primary controllers after a failover event when the primary controller comes back online. This feature is enabled by default, and many administrators choose to leave the AP Fallback default value in place.

However, when an AP falls back to its primary controller, there is a brief window of time—usually 12 to 30 seconds, depending on timer configurations—during which service to wireless clients is interrupted because the APs are rejoining the primary WLC.

If, for some reason, connectivity to the primary WLC has become unstable, the AP may end up flapping back and forth between the primary and backup WLCs. For this reason, many WLAN administrators prefer to disable AP Fallback and move the APs back to the primary controller in a controlled manner during a scheduled service window.

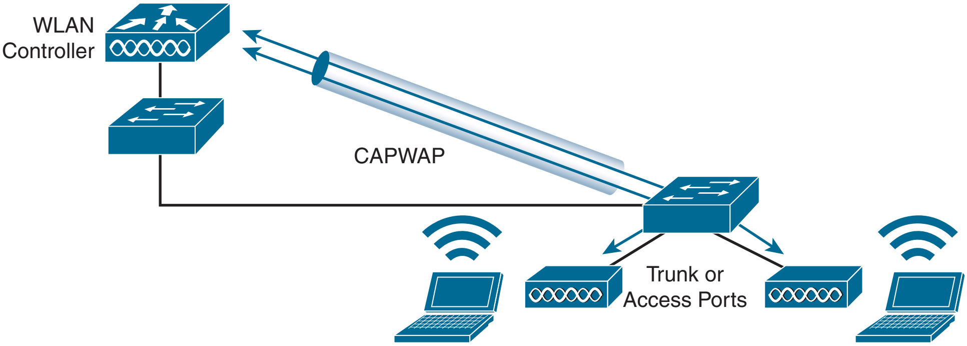
AP Modes

APs that are controlled by a Cisco WLC support different modes of operation. Each mode has unique purpose and properties. Different models of APs support different modes, and not all modes are supported by all APs.

Local Mode

After an AP has discovered and joined its preferred WLC, it typically defaults to the local mode of operation. In local mode, an AP tunnels both management and data traffic to the controller, as illustrated in Figure 11-14. This behavior is known as "centrally switched" because all client traffic is tunneled from the AP to the controller, and the controller is responsible for tagging packets and putting them on the wired network.

Figure 11-14 AP Local Mode



When operating in local mode, an AP allows the monitoring of all channels simultaneously. Off-channel scanning is essential to the operation of a feature called radio resource management (RRM), which gathers information about alternate channel characteristics such as noise and interference. Also, off-channel scanning is responsible for rogue detection.

This activity is accomplished using a 180-second cycle. For a 2.4 GHz radio, this situation means that the AP stays on its assigned channel for 13 seconds, scans the next channel for 50 ms, and returns to its assigned channel for 13 seconds. This process repeats until all channels have been scanned. A similar process occurs with the 5 GHz radio, except that the assigned channel is allowed 10 seconds because of the large number of channels to scan.

FlexConnect Mode

A FlexConnect AP is a wireless solution for branch and remote office deployments using Cisco APs. A FlexConnect AP enables you to configure and control APs in a branch or remote office from the corporate office over a WAN link, without deploying a WLC in each remote location, as illustrated previously in Figure 11-7.

As you saw earlier, FlexConnect APs are capable of supporting several switching modes concurrently, on a per-WLAN basis. The following sections describe these modes.

Locally Switched

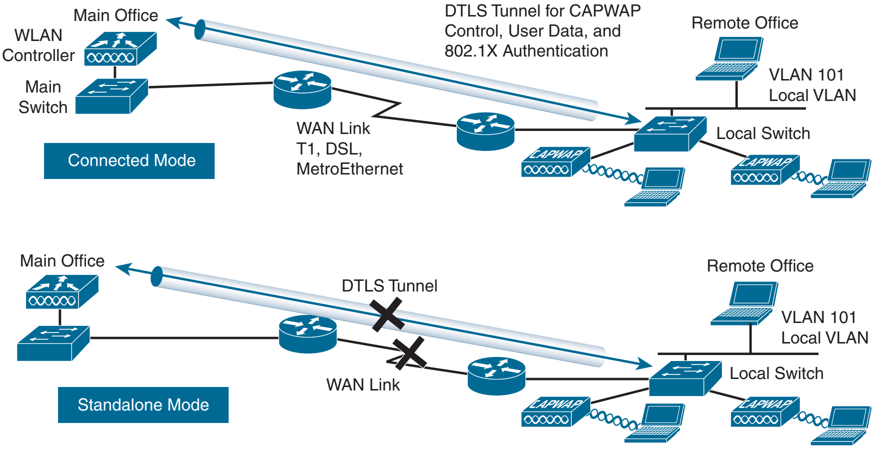
Local switching treats management and data traffic differently. All AP control and management-related traffic is sent to the centralized WLC via CAPWAP. However, local switching keeps branch data traffic local so that it does not consume valuable WAN bandwidth by having to communicate with the central WLC, only to be returned back to the branch.

Centrally Switched

Centrally switched WLANs tunnel both the wireless user traffic and all control traffic via CAPWAP to the centralized WLC, where the user traffic is mapped to a dynamic interface VLAN on the WLC. This is the normal CAPWAP mode of operation, and it is similar to local mode operation.

Cisco FlexConnect functionality varies not only by how each WLAN is configured for data switching (centrally switched or locally switched). FlexConnect functionality also depends on its mode of operation (whether a FlexConnect AP is in connected or standalone mode) and its wireless authentication. Both of these modes are illustrated in Figure 11-15.

Figure 11-15 Cisco FlexConnect Operation


Main Office
WLAN Controller
Main Switch
DTLS Tunnel for CAPWAP Control, User Data, and 802.1X Authentication
Remote Office
VLAN 101 Local VLAN
Local Switch
WAN Link T1, DSL, MetroEthernet
CAPWAP
CAPWAP
Connected Mode
Main Office
DTLS Tunnel
Remote Office
VLAN 101 Local VLAN
Local Switch
WAN Link
CAPWAP
CAPWAP
Standalone Mode

Connected Mode

A FlexConnect AP is said to be in the connected mode when its CAPWAP control plane back to the WLC is up and operational (in other words, the WAN link is not down and the AP and WLC have successfully negotiated a DTLS tunnel across the WAN link). The main difference between an AP in local mode and a FlexConnect AP involves 802.11 authentication and association. All 802.11 authentication and association processing may happen at the FlexConnect AP. The WLC may do all the authentication, may assist with FlexConnect AP proxying to it, or may be out of the loop.

When in connected mode, the FlexConnect AP proxies the associations or authentications to the WLC. In standalone mode, the AP cannot inform the controller of such events. When a client successfully connects, how the AP manages the rest of the session depends on the WLAN configuration and the connection status of the FlexConnect AP.

Standalone Mode

A Cisco FlexConnect AP is said to be in standalone mode when its CAPWAP control plane is not operational, meaning that the WAN link is down, and it no longer has connectivity back to its controller.

Security support on the FlexConnect AP varies. When a FlexConnect AP enters standalone mode, it disassociates all clients that are on centrally switched WLANs because it no longer has a valid path over which to transport their data. Clients authenticated on WLANs using 802.1X authentication and being locally switched remain connected until they roam or until their session times out, unless the AP is provided with a way to reauthenticate the client.

Bridge Mode

A Wi-Fi network can be extended to link LANs. Such a link is referred to as a *bridge*, and these LANs are typically in buildings that lie within a few miles of each other. This linking is the most common use for a Wi-Fi bridge, but there are other uses as well.

Some bridges can be used both to communicate with Wi-Fi clients and to link two networks. Other models are used for bridging purposes only and do not communicate with clients. Cisco Aironet bridges operate at the MAC address layer (data link layer), so they have no routing capabilities. They act as wireless trunks between switches.

A bridge can be point-to-point, simply linking two networks or buildings, or point-to-multipoint, connecting several smaller LANs to a central one. In this topology, as per the role of each bridge, spokes cannot communicate with each other directly but must connect through a central point.

Outdoor networks face specific challenges, such as the impact of humidity and lightning strikes, and specialized help is usually needed for good deployment.

Point-to-point and point-to-multipoint topologies have a downside: If the central point is disconnected or if its bridge is disabled for some reason (such as due to power issues, Ethernet connectivity issues, or radio interference), the spokes are isolated.

In a larger deployment, a network might need to provide connectivity of the spokes not necessarily with a central point but with one another. Such a topology is called a *wireless mesh*. Mesh nodes act as repeaters that transmit data from nearby nodes to peers that are too far away for a manageable cable connection.

Other Modes

Cisco APs support several modes of operation besides those described so far today:

- **OEAP (Office Extend AP) mode:** OEAP is a special type of FlexConnect for teleworkers that provides an Internet connection and establishes a secure tunnel to the corporate network so that remote employees can access applications for a mobility experience that is consistent with the experience in the corporate office. OEAP also allows nonemployees (family members) at the remote location to access the Internet wirelessly without adding extra home devices. These nonemployees can create their own SSIDs, which are routed locally and directly to the Internet.

- **AP monitor mode:** Instead of forwarding client traffic, APs in monitor mode act as dedicated sensors for context-aware (location-based) services, rogue AP detection, and IDS. Two submodes are available: one optimized for RFID tracking and the other for WIPS.
- **AP rogue detector mode:** In rogue detector mode, the AP radio is turned off, and the AP listens to wired traffic only. Because the radio is turned off, it can be placed in the wiring closet, if desired. The rogue detector AP listens for ARP packets on the wire and caches them. This address cache is used to determine if the Layer 2 addresses of an identified rogue client or AP are present on the wire.
- **AP sniffer mode:** With sniffer mode, an AP can be placed into promiscuous mode and can capture all IEEE 802.11 transmissions that it receives. The packets, including information on timing and signal strength, are forwarded to a remote PC that runs a network analyzer software package such as OmniPeek, Wireshark, or AirMagnet.
- **AP SE-Connect mode:** SE-Connect mode, also referred to as SOMM (Spectrum-Only Monitor Mode), allows any Cisco CleanAir AP to be configured as a network-connected sensor. This sensor gathers information on the signal strength and duty cycle of every RF transmission within the bands that are utilized by the wireless network. This raw spectrum information is then forwarded to a workstation running MetaGeek Chanalyzer with CleanAir or Cisco Spectrum Expert for analysis.

Antenna Characteristics

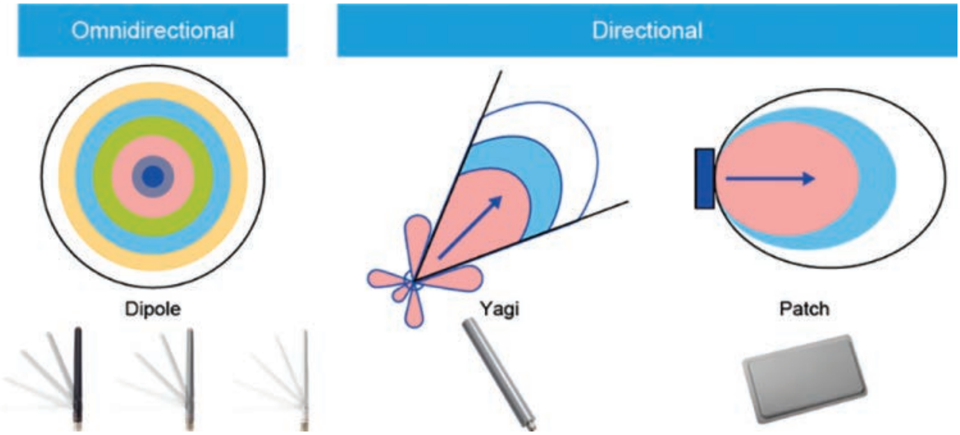
This section describes the various antennas and the principles of transmission. An antenna is needed to transmit RF signals. Several types of antennas are available. Which antenna you decide to use depends on where and how you want the signal to be received.

Antenna Types

The two main families of antennas are omnidirectional and directional. An omnidirectional antenna sends a signal of the same strength in all directions (although an antenna can never be completely omnidirectional). A directional antenna concentrates the beam in one or more directions. Figure 11-16 shows these two categories of antenna types.

A directional antenna radiates the same amount of energy as an omnidirectional antenna. The main difference is that a directional antenna focuses the beam in a specific direction. Because a directional antenna is stronger in a specific direction, it is said to add more *gain*. In other words, a directional antenna is more powerful in that specific direction than an omnidirectional antenna. In addition, the rating of the directional antenna, in dBi, is higher than it would be for an omnidirectional antenna.

Figure 11-16 Antenna Types



One way of representing the change in radiation pattern is to talk about angles. An omnidirectional antenna radiates all around itself equally, or at 360 degrees. A directional antenna radiates toward a certain direction, but the beam might be wide or narrow; its angle is said to be more or less narrow.

The angle for each antenna type is fixed and is shown in the radiation pattern. This angle is called the *beamwidth*, and it correlates directly to the strength of the signal in the main beam: The narrower the beam, the higher the strength of the signal in the beam and, therefore, the higher the gain. Some antennas are said to be high gain because they concentrate the beam, allowing these antennas to reach faraway points with a good signal.

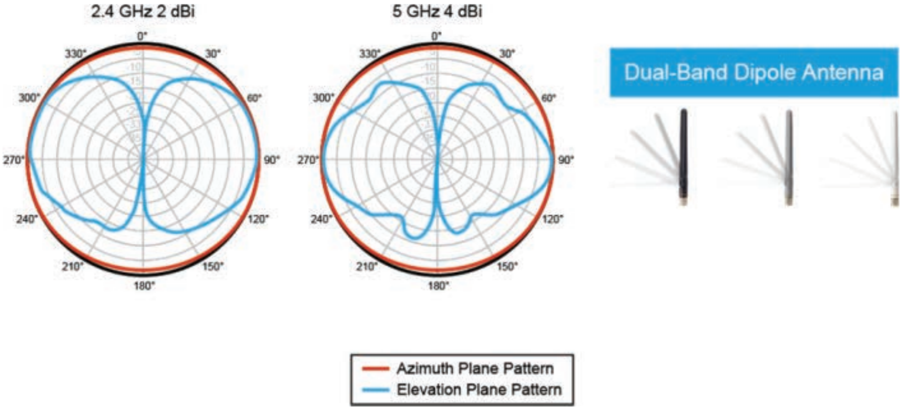
Omnidirectional Antennas

An omnidirectional antenna radiates in a three-dimensional (3D) environment, but providing a 3D view of the radiation pattern is possible only if the person who intends to purchase the antenna has 3D-capable viewing software. Therefore, vendors usually provide two views:

- **Horizontal plane (H-plane, or azimuth chart):** The H-plane represents the radiation pattern, as seen from the top. This chart shows how the signal spreads ahead, behind, to the right, and to the left but not how the signal spreads up or down. This chart provides a top-down view.
- **Elevation plane (E-plane):** The E-plane, or elevation chart, represents the radiation pattern as seen from the side of the antenna. This chart shows how the signal spreads ahead, behind, to the top, and to the bottom but not how the signal spreads to the right or left. This chart provides a side view.

Omnidirectional antennas are omnidirectional at least on one plane, usually the azimuth plane. They radiate around themselves, as seen from the top, but do not necessarily do so in the vertical plane, as illustrated in Figure 11-17. This design makes the shape of the radiation pattern more like a donut than a pure sphere. This shape is common for many indoor omnidirectional antennas, though the thickness of the donut varies.

Figure 11-17 Omnidirectional Antenna



The most basic omnidirectional antennas are dual-band dipole antennas. This type of antenna is not very powerful and is typically designed for an indoor environment with APs or client cards. The antenna radiates everywhere on the H-plane to reach clients in the whole room or surrounding premises; however, the antenna has a certain vertical angle because it is expected to work on a "one-floor coverage" logic in which the AP destination is on the same floor as the antenna.

For most office environments, you will probably use APs with integrated antennas. These antennas are omnidirectional and have a gain of 4 dBi for both radio bands.

When you need to cover a large area, a 2 or 4 dBi antenna might not provide enough gain. For instance, these antennas might not reach all the corners of a large conference room. An antenna that might meet this need is a high-gain omnidirectional ceiling mount antenna, which provides a 5.2 dBi gain. This antenna is typically mounted on a ceiling (pointing downward to respect polarity) to cover a large hall.

Directional Antennas

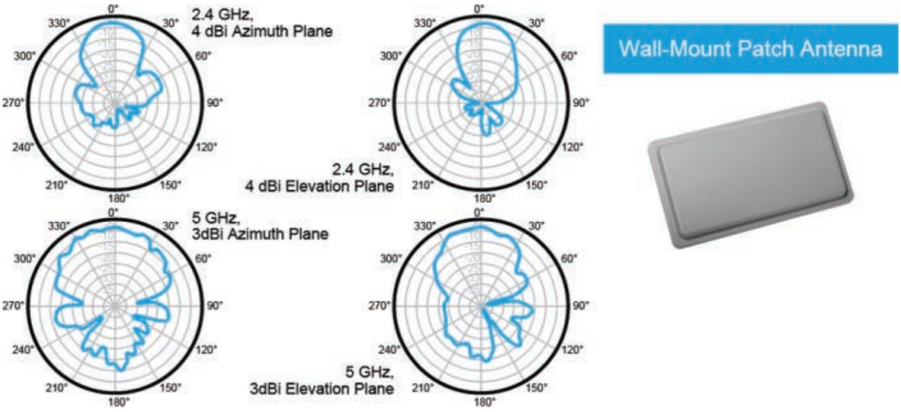
A directional antenna is designed to cover one specific direction. In an indoor environment, these antennas are typically placed on walls to provide coverage in a long rectangular room.

A directional antenna provides more gain than a basic omnidirectional dipole antenna, resulting in a 3 to 4 dBi directional gain instead of a 2.14 dBi omnidirectional gain.

The shape of the antenna coverage pattern shows that it follows one-floor coverage logic, in which the beam does not have a large vertical range.

Figure 11-18 shows a patch antenna, which is ideal for an indoor environment because it is flat. The appearance of this antenna is discrete and does not draw attention. The antenna radiates slightly toward the back. This backward radiation is useful when positioning this kind of antenna over doors; access is possible when users reach the door. If the antenna is positioned on a wall without a door, the wall absorbs most of the backward signal.

Figure 11-18 Directional Antenna



Another type of directional antenna is the Yagi or Yagi Ude (so called because it was invented by H. Yagi and S. Ude in Japan in 1926). This antenna, with its comb shape, is the same type that is used for TV reception. In Wi-Fi networks, the active comb is usually encased in a protective cylinder.

The pattern of this antenna is specific: rather thin on the vertical and horizontal planes and reaching far in a narrow beam. The antenna has slight radiation at the back due to the manufacturing process. Most Yagis have this small radiated field at the back, which is called the butterfly effect. The manufacturing process also creates side lobes: thin coverage areas to the sides of the main field.

The gain of this antenna is high, at 13.5 dBi. The antenna is well adapted to covering long corridors or large warehouses (if several Yagis are installed next to one other).

Another example of a classical directional antenna is the parabolic dish. Its shape makes this antenna adaptable to long-range outdoor links, mainly in a point-to-point scenario. Because the beam is so focused, the gain is very high. The 21 dBi gain makes this antenna almost 100 times as powerful as a dipole antenna.

Study Resources

For today's exam topics, refer to the following resource for more study.

Resource	**Module, Chapter, or Link**
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	18

This page intentionally left blank

Day 10

Wireless Client Roaming and Authentication

ENCOR 350-401 Exam Topics

Architecture

- Analyze design principles of a WLAN deployment
  - Location services in a WLAN design

Infrastructure

- Wireless
  - Describe the main principles and use cases for Layer 2 and Layer 3 roaming

Security

- Configure and verify wireless security features
  - EAP
  - WebAuth
  - PSK

Key Topics

Today we continue reviewing wireless topics, with a particular focus on client roaming and authentication.

As a wireless client moves out of the effective range of one access point, it must associate with a new access point to maintain connectivity with the network. Mobility groups enable wireless LAN controllers to make handoffs between APs quite seamlessly. Today we look at how roaming is triggered and the difference between Layer 2 and Layer 3 roaming, using one or more controllers. In addition, we explore how Cisco Connected Mobile Experiences (CMX) provides location services to track client movement and to provide valuable analytic data.

Security is a primary concern in modern networks. Security is particularly important in WLANs, in which the radio signal can spread beyond the edge of the building and the security of the building's physical premises. An attacker outside the enterprise perimeter might be able to detect a valid client signal and even associate with a corporate access point.

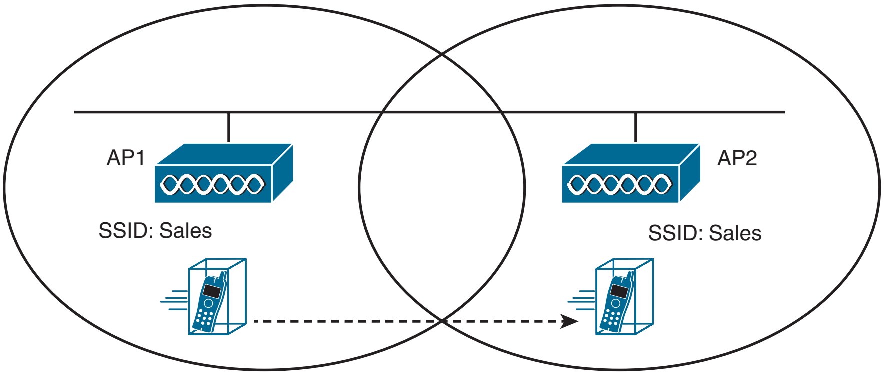
Today we discuss various wireless security components and explore how 802.11 works with 802.1X and the Extensible Authentication Protocol (EAP) framework to provide wireless client authentication. We look at how each of these security methods provides a secure WLAN for employees while also providing guest access for the various devices that are common in the enterprise environment.

Wireless Roaming

A wireless client typically decides to roam to a different AP when degradation occurs in its current connection to an AP. Roaming necessarily has some impact on client traffic because while a client scans other channels for alternative APs, reassociates, and authenticates to the new AP, it may not be able to transmit traffic.

Figure 10-1 shows a wireless IP phone roaming from AP1 to AP2 as the user physically moves from one coverage area to the next. Notice that the service set identifier (SSID) remains the same from AP1 to AP2.

Figure 10-1 Wireless Roaming



Although the roaming algorithms differ for each vendor or driver version (and potentially for different device types from a single vendor), there are some common situations that typically cause roaming:

- **Maximum data retry count is exceeded:** Excessive numbers of data retries commonly trigger roaming. Each vendor has a different threshold, which is linked to the data rate. A given threshold triggers a shift to a lower data rate and another threshold to roam.
- **Low received signal strength indication (RSSI):** A client device can decide to roam when the received signal strength drops below a threshold. This roaming trigger does not require active client traffic to induce roaming.
- **Low signal-to-noise ratio (SNR):** A client device can decide to roam when the difference between the noise floor drops rises above a certain threshold. This roaming trigger does not require active client traffic to induce roaming.
- **Proprietary load-balancing schemes:** Some wireless implementations have schemes in which clients roam to more evenly balance client traffic across multiple APs. In such a case, roaming might be triggered by a decision in the WLAN infrastructure and communicated to the client via vendor-specific protocols. Cisco wireless LAN controllers (WLCs) are configured

with a default set of radio frequency (RF) roaming parameters, which are used to set the RF thresholds that are adopted by the client to decide when to roam. It is possible to override the default parameters by defining a custom set of parameters. These Cisco Compatible Extensions (CCS) parameters are defined on the controller for each frequency band.

Wireless clients learn about available APs by scanning other 802.11 channels for available APs on the same WLAN or SSID. Other 802.11 channels can be scanned actively or passively:

- **Active scan:** Active scanning occurs when a client changes its 802.11 radio to the channel that is being scanned, broadcasts a probe request, and then waits to hear any probe responses (or periodic beacons) from APs on that channel (with a matching SSID). The 802.11 standards do not specify how long the client should wait, but 10 ms is a typical period. The probe-request frames in an active scan can be one of two types**:**
    - **Directed probe:** The client sends a probe request with a specific destination SSID; only APs with a matching SSID reply with a probe response.
    - **Broadcast probe:** The client sends a broadcast SSID (actually a null SSID) in the probe request; all APs receiving the probe request respond with a probe response for each SSID they support.
- **Passive scan:** Passive scanning is performed by simply changing the 802.11 radio of the client to the channel that is being scanned and waiting for a periodic beacon from any APs on that channel. By default, APs send beacons approximately every 100 ms. Because it can take up to 100 ms to hear a periodic beacon broadcast, most clients use active scanning instead.

During a channel scan, the client is unable to transmit or receive client data traffic. Clients take various approaches to minimize the impact to their data traffic:

- **Background scanning:** Clients can scan available channels before they need to roam. Such scans build knowledge of the RF environment and available APs so clients can roam faster if doing so becomes necessary. The impact on client traffic can be minimized by scanning only when the client is not actively transmitting data or by periodically scanning only a single alternate channel at a time (because scanning a single channel incurs minimal data loss).
- **On-roam scanning:** In contrast to background scanning, on-roam scanning occurs when roaming is necessary. Each vendor and each device can implement its own algorithms to minimize the roam latency and the impact on data traffic. For example, some clients might scan only the nonoverlapping channels.

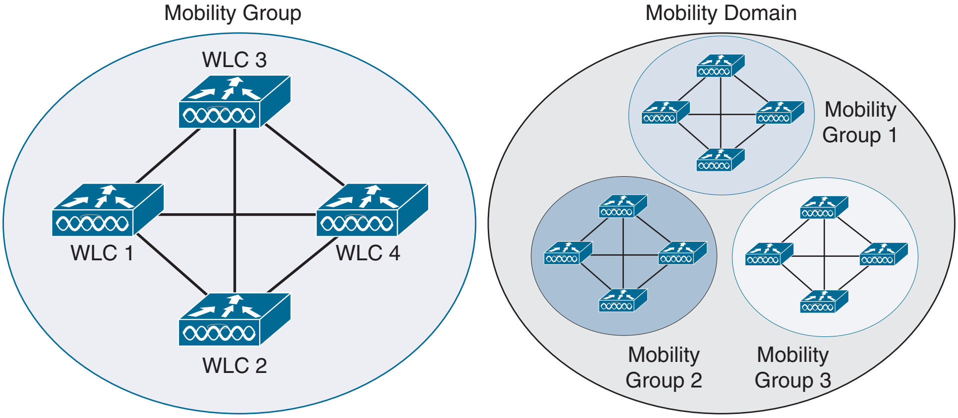
Mobility Groups and Domains

A mobility group is a collection of mobility controllers (MCs) across which roaming needs to be supported. A mobility group can consist of up to 24 WLCs that share the context and state of client devices and WLC loading information. The WLCs of a mobility group forward data traffic among the group members, which enables inter-controller WLAN roaming and WLC redundancy.

A wireless client can roam between any APs in a mobility group without being required to reauthenticate to the network. If the AP being roamed to is not on the same WLC as the AP being roamed from, the client information is transferred.

A controller can recognize controllers that belong to another mobility group. When this occurs, the controllers are said to be in the same *mobility domain*. Up to three separate mobility groups can be cross-populated into each other's mobility lists to form a mobility domain that supports up to 72 controllers. Client roaming can occur between controllers that are in different mobility groups as long as the controllers are in the same mobility domain. Figure 10-2 illustrates the concept of mobility groups and mobility domains.

Figure 10-2 Wireless Mobility Groups and Mobility Domains



For two controllers to be in the same mobility domain, they must know each other, which means that the built-in MAC address and management IP address of each controller must be entered in the other controller. To transmit mobility control messages, Cisco WLCs use UDP port 16666 for unencrypted traffic. User data traffic is transmitted via Ethernet over IP (EoIP) using IP protocol number 97 or CAPWAP (UDP port 5246) tunnels.

If a client moves from one controller to another in a different mobility domain (that is, to a controller that is not known by the controller that the client leaves), the client has to reauthenticate, reassociate, and get new IP address information.

The way mobility groups are organized in a wireless network determines whether and how roaming occurs. To ensure fast, secure roaming, all controllers where roaming should occur should be in the same mobility group. This logical grouping is usually determined after a site survey.

Wireless Roaming Types

As the client moves from AP to AP, the controller manages all roaming activity, and the client is not affected. Even when a client roams to a different controller, it is not affected and does not even recognize the roaming event. This seamless roaming process implies that both controllers are in the same mobility group or domain.

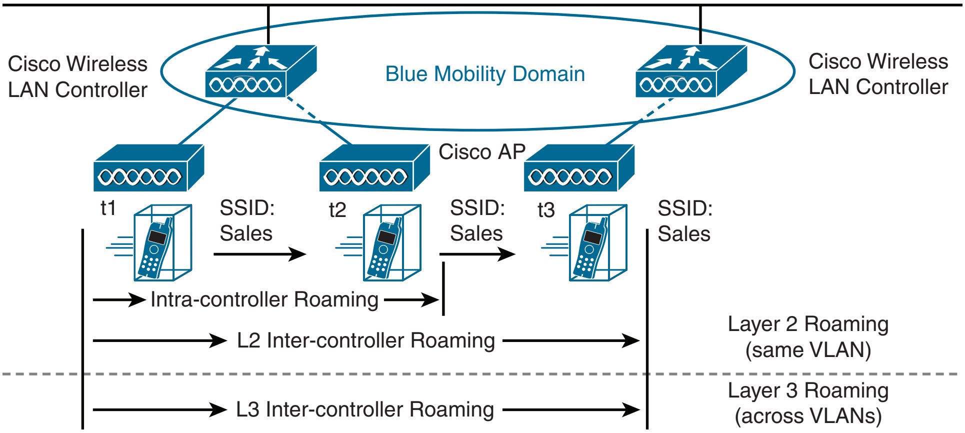
When a wireless client associates with and authenticates to an AP, the AP controller places an entry for this client in its client database. This entry includes the following information:

- MAC and IP addresses of the client
- Security context and associations

- QoS contexts
- The WLAN
- The associated AP

The controller uses this information to forward frames and manage traffic to and from the wireless client. There are two types of wireless client roaming—Layer 2 roaming and Layer 3 roaming—as illustrated in Figure 10-3.

Figure 10-3 Wireless Roaming


Cisco Wireless LAN Controller
Blue Mobility Domain
Cisco Wireless LAN Controller
Cisco AP
t1
SSID: Sales
t2
SSID: Sales
t3
SSID: Sales
Intra-controller Roaming
L2 Inter-controller Roaming
Layer 2 Roaming (same VLAN)
L3 Inter-controller Roaming
Layer 3 Roaming (across VLANs)

Layer 2 Roaming: Centralized Controllers

Layer 2 roaming occurs when a client moves from one AP to another and is maintained in the same client subnet/VLAN. This type of roaming can occur in two ways:

- **Intra-controller roaming:** When a client moves from location t1 to t2 in Figure 10-3, it asks for reauthentication on a new AP. This authentication is done by a query that is sent to the controller to which the AP is connected. If the controller is the same one to which the AP that the client is leaving was associated, the controller simply updates the client database with the newly associated AP. If necessary, new security context and associations are established as well. This internal operation, known as intra-controller Layer 2 roaming, is the simplest and quickest type of roaming.
- **Inter-controller roaming:** In this process, a client roams from an AP that is joined to one controller to an AP that is joined to a different controller. When the client at location t2 moves to t3, it associates to an AP that is joined to a new controller, the new controller exchanges mobility messages with the original controller, and the client database entry is copied to the new controller. New security context and associations are established, if necessary, and the client database entry is updated for the new AP.

In Layer 2 roaming, it is assumed that the client will remain in the same subnet; no Layer 3 activity is required. Layer 2 roaming assumes that any WLCs that are involved in the roaming are Layer 2 adjacent, share the same VLAN information, and are in the same mobility group. They share SSIDs that are mapped to the same VLAN and subnet. The controllers should also have a compatible version of AireOS code, and in centralized deployments, the controller is both the mobility controller (MC) and the mobility agent (MA). The MC handles roaming, radio resource management (RRM), and AP licenses. The MA terminates CAPWAP tunnels and maintains the client database.

Layer 2 Inter-Controller Roaming

Figure 10-4 shows a client's connectivity before roaming, and Figure 10-5 shows the client's connectivity after roaming.

Figure 10-4 L2 Inter-Controller Wireless Connectivity: Before Roaming

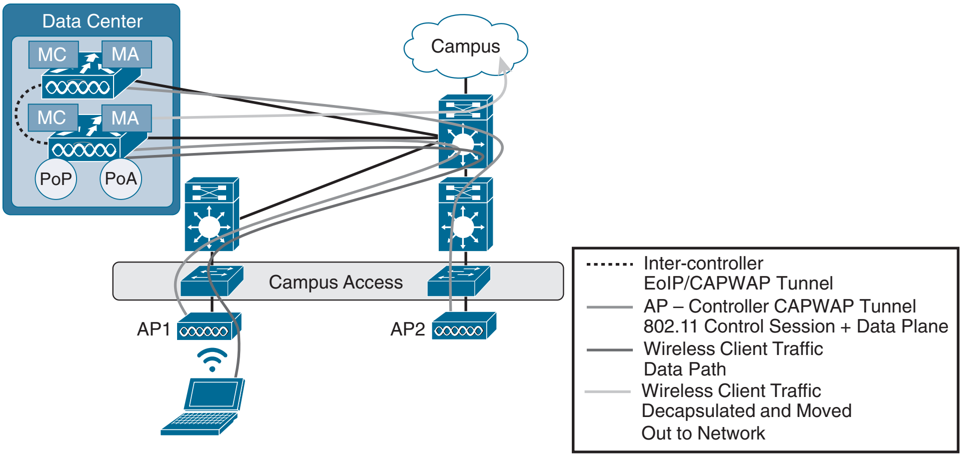
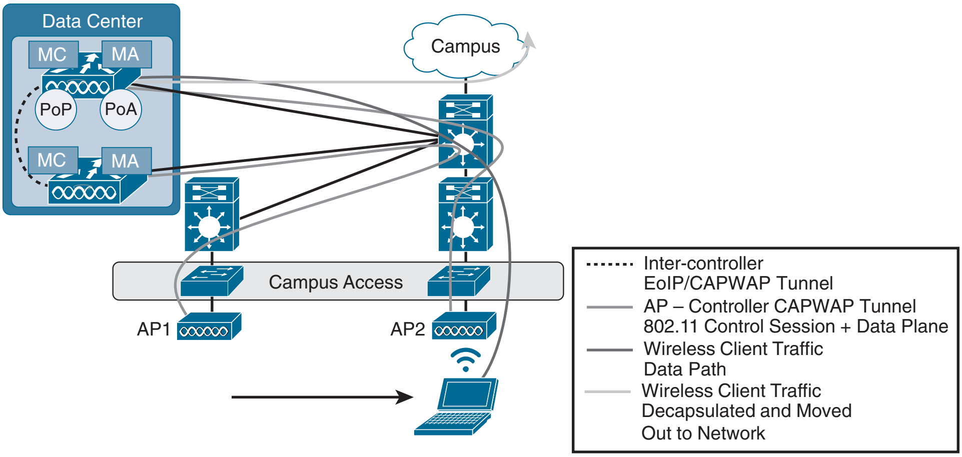

Data Center
MC
MA
MC
MA
PoP
PoA
Campus
Campus Access
AP1
AP2
Inter-controller EoIP/CAPWAP Tunnel
AP – Controller CAPWAP Tunnel 802.11 Control Session + Data Plane
Wireless Client Traffic Data Path
Wireless Client Traffic Decapsulated and Moved Out to Network

Figure 10-5 L2 Inter-Controller Wireless Connectivity: After Roaming


Data Center
MC
MA
PoP
PoA
MC
MA
Campus
Campus Access
AP1
AP2
Inter-controller EoIP/CAPWAP Tunnel
AP – Controller CAPWAP Tunnel 802.11 Control Session + Data Plane
Wireless Client Traffic Data Path
Wireless Client Traffic Decapsulated and Moved Out to Network

The Layer 2 roaming steps are as follows:

1. Initially, the point of presence (PoP) and the point of attachment (PoA) of the user are co-located on the same controller. The PoP is where the wireless user is seen to be within the wired portion of the network. It anchors the client IP address and is used for security policy application. The PoA is where the wireless client has roamed to in the network. The PoA moves with user AP connectivity and is used for QoS policy application.
2. The user roams from AP1 to AP2 and its associated controller.
3. The new controller exchanges mobility messages with the original controller, and the client database entry is moved to the new controller (assuming that the controllers are part of the same mobility group).
4. New security context and associations are established, if necessary, and the client database entry is updated for the new AP.
5. The entire mobility context (PoP and PoA) of the client is moved.
6. The client traffic is also moved to the new path.

All this is transparent to the client if there is no DHCP discovery request or if a session timeout is exceeded.

Layer 3 Roaming: Centralized Controllers

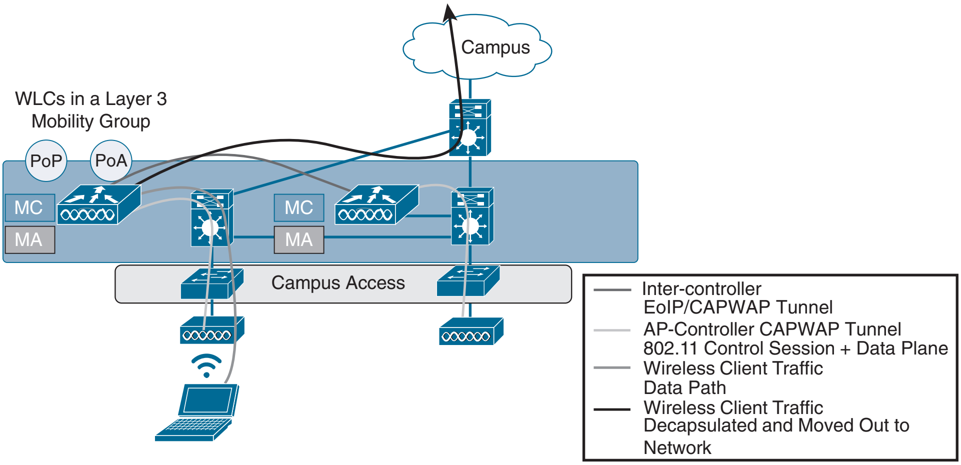
Layer 3 roaming occurs when a client moves from an SSID on one AP (that is associated with one VLAN and respective IP subnet) to the same SSID on a different AP (that is associated with a different VLAN and IP subnet). This type of roaming can occur in the following ways:

- **Inter-controller roaming:** Roaming from one subnet to another is synonymous with inter-controller roaming in that the controllers exchange mobility messages about the client roaming. However, instead of moving the client database entry to the new controller, the original controller marks the client with an anchor entry in its own client database. The database entry is copied to the new controller client database and marked with a foreign entry in the new controller. The roaming remains transparent to the wireless client, and the client maintains its original IP address.
- **Inter-subnet roaming:** With this type of roaming, WLANs on both anchor and foreign controller(s) need to have the same network access privileges and no source-based routing or source-based firewalls in place. Otherwise, the clients may have network connectivity issues after the handoff.

Layer 3 Inter-Controller Roaming Example

In Figure 10-6, the two controllers do not share a common set of backend VLANs. They are sharing the same SSIDs; however, these SSIDs are mapped to different VLANs on their respective switches. Because the distribution switches to which the controllers are attached are separated by a routed core, the traffic does not flow across the same forwarding VLANs.

Figure 10-6 L3 Inter-Controller Wireless Roaming: Before


Campus
WLCs in a Layer 3
Mobility Group
PoP
PoA
MC
MA
MC
MA
Campus Access
Inter-controller
EoIP/CAPWAP Tunnel
AP-Controller CAPWAP Tunnel
802.11 Control Session + Data Plane
Wireless Client Traffic
Data Path
Wireless Client Traffic
Decapsulated and Moved Out to
Network

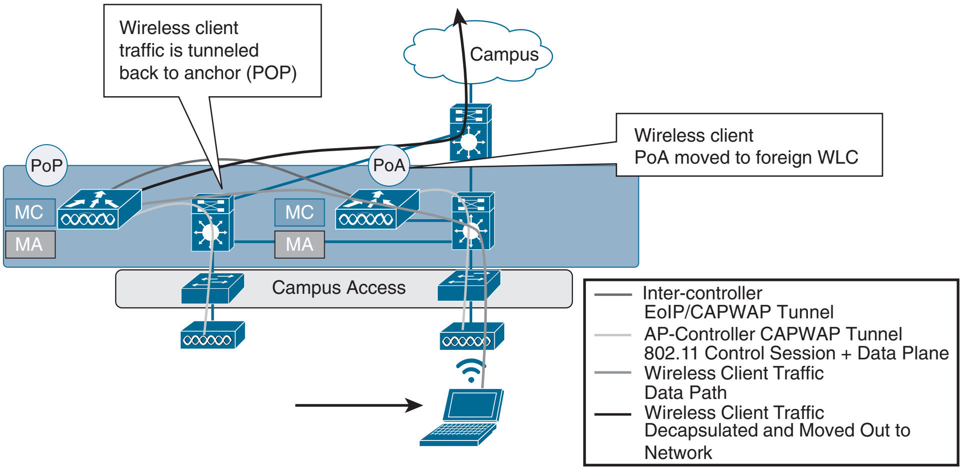
Because the new WLC does not share the same subnet information, the client loses its IP address unless it was accomplished through Layer 3 roaming. This condition requires that the PoP of the client be anchored to the original WLC and that the PoA be moved to the foreign WLC.

When a wireless client associates to an AP, the controller that is associated with the AP places an entry for this client in its client database. This entry includes the MAC and IP addresses, security context and associations, QoS contexts, WLAN, and associated AP of the client. The controller uses this information to forward frames and manage traffic to and from the wireless client. In this Layer 3 roaming scenario, there are two controllers that are grouped in a common mobility group.

The Layer 3 inter-controller roaming steps are as follows:

1. Initially, the PoP and PoA of the user are co-located on the same controller.
2. The user roams from one AP and its associated controller (anchor) to another AP and its associated controller (foreign). This is shown in Figure 10-7.
3. The new controller exchanges mobility messages with the original controller, and the client database entry is copied to the new controller (in the same mobility group).
4. A new security context and associations are established, if necessary, and the client database entry is updated for the new AP.
5. The client PoA is moved to the new controller (foreign).
6. The client PoP stays fixed to the original controller (anchor). (This is done to ensure that the user retains the same IP address across Layer 3 boundary roaming and also to ensure continuity of policy application during roaming.)
7. The client traffic is tunneled back to the original controller (anchor). This is called *symmetric mobility tunneling*.

Figure 10-7 L3 Inter-Controller Wireless Roaming: After


Wireless client
traffic is tunneled
back to anchor (POP)
Campus
Wireless client
PoA moved to foreign WLC
PoP
PoA
MC
MA
MC
MA
Campus Access
— Inter-controller EoIP/CAPWAP Tunnel
— AP-Controller CAPWAP Tunnel 802.11 Control Session + Data Plane
— Wireless Client Traffic Data Path
— Wireless Client Traffic Decapsulated and Moved Out to Network

Roaming with Auto-Anchor Mobility (Guest Access)

Auto-anchor mobility, also referred to as *guest tunneling*, is a subset of a mobility group in which all the client traffic that belongs to a WLAN (typically the Guest WLAN) is tunneled to a predefined WLC or set of controllers configured as an anchor for that specific WLAN. This feature helps restrict clients to a specific subnet and provides more control over the user traffic.

The auto-anchor process is as follows:

1. The wireless client associates to an AP on WLC1 on a guest SSID, as illustrated in Figure 10-8.
2. The guest user PoA is with WLC1, through which it initially associated.
3. WLC1 is pointed to the guest anchor for the guest SSID.
4. The guest user traffic is tunneled to the guest anchor WLC at the top. (Remember that this symmetric tunnel is EoIP or CAPWAP.)
5. The guest user IP comes from the guest anchor.
6. The guest anchor becomes the PoP for the guest and stays static.
7. The guest, once authenticated (assuming WebAuth on the guest anchor), has its traffic decapsulated at the anchor WLC and moved out to the network (to the firewall and then to the Internet).

Because the guest client is anchored to the guest anchor WLC, its authentication and IP addressing are sourced at this location. The guest client IP address, the guest client PoP, and the guest anchor do not change as the client roams, as shown in Figure 10-9.

Figure 10-8 Guest Roaming with Auto-Anchor Mobility: Before

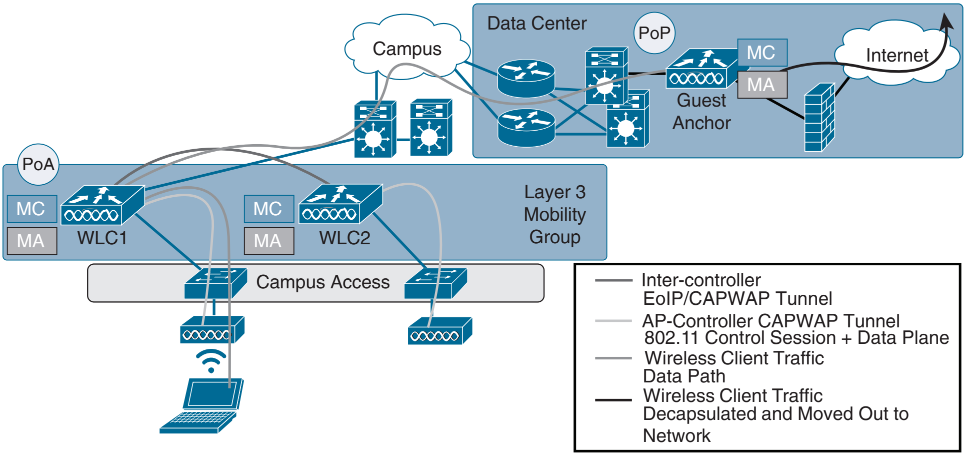
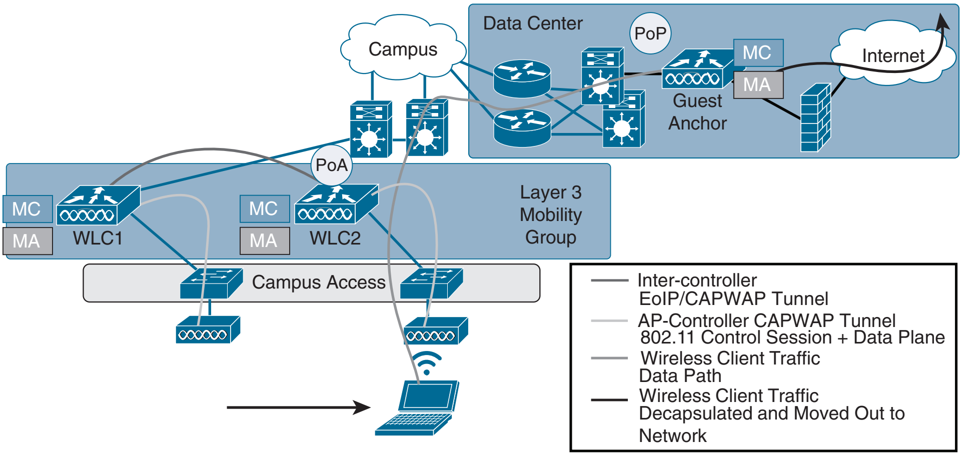

Campus
Data Center
PoP
MC
MA
Internet
Guest Anchor
PoA
MC
MA
WLC1
MC
MA
WLC2
Layer 3 Mobility Group
Campus Access
Inter-controller EoIP/CAPWAP Tunnel
AP-Controller CAPWAP Tunnel 802.11 Control Session + Data Plane
Wireless Client Traffic Data Path
Wireless Client Traffic Decapsulated and Moved Out to Network

Figure 10-9 Guest Roaming with Auto-Anchor Mobility: After


Campus
Data Center
PoP
MC
MA
Internet
Guest Anchor
PoA
MC
MA
WLC1
MC
MA
WLC2
Layer 3 Mobility Group
Campus Access
Inter-controller EoIP/CAPWAP Tunnel
AP-Controller CAPWAP Tunnel 802.11 Control Session + Data Plane
Wireless Client Traffic Data Path
Wireless Client Traffic Decapsulated and Moved Out to Network

Wireless Location Services

Research and development in Wi-Fi location prediction techniques has facilitated the emergence of indoor RF location tracking systems based fundamentally on the Institute of Electrical and Electronics Engineers (IEEE) 802.11 infrastructure. (This should not be confused with solutions requiring a dedicated, independent infrastructure of location receivers and radio frequency identification [RFID] tag readers.) Location-based services are used to perform network analytics and make it possible to collect meaningful data about the wireless aspect of a network.

Cisco CMX

Cisco Connected Mobile Experiences (CMX) is a tool that provides leveraged analytics to understand mobile device patterns. CMX allows a wireless network to deliver dynamic and relevant content to mobile end users. Once a consumer connects to the network, the network provides the organization with many different ways to engage with the consumer.

Cisco CMX has three features:

- **Detect:** With the Detect feature, Cisco CMX can provide either presence or location-based detection using the Wi-Fi signal. Presence is for smaller locations with only one or two APs but where you still want to determine the number of visitors and the time they are spending at a site. The time that the devices spend in a venue is referred to as *dwell time*. You can also use presence to see how many devices have joined the network as opposed to being a passerby. You can also determine how long the devices have been on the network or in the venue, and those devices can be tracked over different time periods, which makes it possible to make comparisons that are based on days, weeks, or months.

  Using location detection provides more detailed capabilities. Location-based detection depends on the type and number of APs that you have in your venue. It can provide location accuracy from 10 meters all the way down to 1 meter of accuracy. The ability to provide 1 meter of accuracy is based on the Cisco access points with the Hyperlocation module and antenna array attached. You can also provide a location that is based on Bluetooth Low Energy (BLE), which requires BLE modules to be installed in the venue to detect devices through BLE.

- **Connect:** The Connect feature allows Cisco CMX to provide a customized portal for guests to log in to the network. Such a portal can be customized to provide a splash screen when the guests associate with the network. The guests need to enter their information, and they might have to agree to an acceptable use policy (AUP) before they can be granted access to the network. The portal can be configured to support a password to allow the guest to be authenticated with the network. The password could be sent to the guest through an email account or through an SMS.

  Another way to authenticate a guest is by using a social media account. For example, guests may be able to use their Facebook, Instagram, or Twitter accounts. Using a social media service for authentication requires the organization to set up and configure accounts so that the end users can use OAuth for authentication. The benefit of this form of authentication is that the venue can extract more analytics information from the social media accounts about the guests. The portal can also be configured for a specific area in the network (zone). For example, you might want to set up the portal in the zones that cover the entrances to your retail stores. Once a guest has provided information, Cisco CMX begins to collect the presence or location data.

- **Engage:** The Engage feature allows Cisco CMX to provide content to guests. For example, if a guest enters the venue and is a recurring guest, you might send a coupon of some type. As another example, a retail store may have a customer loyalty program, and a guest who is a member of the program will be sent a different coupon than a regular guest, or the guest may get additional loyalty points for purchases. In a hospitality scenario, a loyalty member, based on the level of membership (for example, diamond, platinum, gold, or silver), might receive better Wi-Fi service while in the hotel.

Cisco CMX can be deployed as either an on-premises or cloud-based solution. CMX on-premises software can be deployed on a preinstalled Cisco Mobility Services Engine (MSE) appliance or as a virtual machine on a generic server such as a Cisco Unified Computing System (UCS) appliance. CMX Cloud has two SaaS licensing tiers, and both of them include software and technical support.

Cisco CMX Analytics Tools

Cisco CMX includes two tools that provide useful information for making make important business decisions: Presence Analytics and Location Analytics.

Presence Analytics

Presence Analytics is a tool for counting devices. Depending on your settings in Cisco CMX, you may have one or more floors. The number of guests can be counted by floor or, in a larger sense, by building. The time frame can be selected so that you can look at only the information that is relevant for your needs.

You can also see the data for repeat vs. new visitors, and you can look at the number of visitors vs. passersby and the conversion rate from passerby to visitor. The conversion rate from passerby to visitor can be the most important information if you are rolling out a new marketing plan to entice new visitors into the store. After the visitors enter the store, you can track the amount of time they spend in the store, which is known as the *dwell time*.

For example, you might want to determine whether new or repeat visitors spend more time in the store. Because you have the data about the number of visitors in the store, you can also monitor the busiest hours of the day and the busiest days of the week. This data can be compared with historical data from past days or weeks to determine whether more visitors are coming into the store and at what time or day of the week. Such analysis is helpful in managing store staff so that the visitors have a great experience while in the store.

Location Analytics

Cisco CMX Location Analytics allows you to look at heatmaps, which provide visual representations of total activity on floor maps in real time and can also be viewed as playbacks over time. This is very useful for understanding the space utilization and popularity of different sections of a location or zone.

Correlation is used with location-based services, which allows Cisco CMX to track the movement of guests through the venue. For example, say that you operate a mall and would like to see where guests are coming from to enter the food court. Cisco CMX can track the movements of guests in the food court, too. When the guests leave the food court, where do they go in the mall? Do they continue shopping, or do they leave the mall? Thanks to Location Analytics, Cisco CMX can show these paths.

Location Accuracy

How precise you are with locating mobile devices depends on the manner in which the devices are found.

If you want to provide presence, including device count and dwell time, you can accomplish this with a single AP. The accuracy is 20 meters, which is a fairly large area. However, if you are trying to determine the number of devices in a venue, the accuracy does not need to be as high as when you are trying to determine whether the devices are in a specific area of a venue.

To improve the accuracy, you need at least three APs to hear client traffic. Note that the traffic can be data or probes. Depending on the size of the venue, more than three APs may be necessary to provide good coverage, which will provide 7 meters of accuracy.

To achieve 5 meters of accuracy, you need to enable FastLocate, and you also need to increase the density of the APs. FastLocate adds the ability to get RSSI or location information using data packets that the AP receives for higher location refresh rates. Using the data packet, location-based service updates are initiated by the network and are available more frequently, which provides more data points to accurately represent end-user activity.

While it is tempting to think of capturing RSSI from data packets as an incremental improvement over capturing RSSI from probe frames, it is actually a significant step forward. When Wi-Fi mobile devices connect to one AP, neighboring APs are unable to hear the data packets because they are on different channels, which is bad for location estimation. FastLocate technology uses the wireless security modules (WSMs) available in certain Cisco APs. The WSMs in the associated AP and the neighboring APs scan all channels at the same time to get multiple RSSI readings from a data packet transmission and, thus, locate the mobile device. This process takes place without taxing the data-serving radios and has no performance hit to the data service on the serving radios in the 2.4 and 5 GHz bands.

To achieve 1 meter of accuracy, you need Cisco APs that support the Cisco Hyperlocation antenna and module. Cisco Hyperlocation is an enhanced location solution that includes Wi-Fi-based location to determine the location of mobile devices and BLE technologies to act as a beacon for customer interactions.

Wireless Client Authentication

When the Wired Equivalent Privacy (WEP) standard was found to be weak and easily breakable, both the IEEE 802.11 committee and the Wi-Fi Alliance worked to replace it. Two generations of solutions emerged: Wi-Fi Protected Access (WPA) and its successor, WPA2. These solutions offer a security framework for authentication and encryption.

WPA2 is the current implementation of the 802.11i security standard and deprecates both WEP and WPA. WPA2 being 802.11i compliant is the current standard for enterprise networks. Unlike WPA, WPA2 provides support for IEEE 802.11n/ac. WPA2 provides either 802.1X or PSK authentication and determines two modes of wireless protected access:

- WPA2 Personal mode
    - Uses WPA2-PSK (Pre-Shared Key) authentication, with which a common key is statically configured on the client and the AP
    - Designed for environments where there is no Remote Authentication Dial-In User Service (RADIUS) authentication server
    - Provides inadequate security for an enterprise wireless network because if attackers break the WPA2-PSK, they can access the network and user data

- WPA2 Enterprise mode
    - Uses IEEE 802.1X and Extensible Authentication Protocol (EAP) authentication so that each user or device is individually authenticated
    - Incorporates a RADIUS authentication server for authentication and key management
    - Used by enterprise-class networks

In 2018, the Wi-Fi Alliance announced WPA3 as a replacement for WPA2. WPA3 is available in Personal and Enterprise modes. WPA3 replaces WPA2-PSK with Simultaneous Authentication of Equals (WPA3-SAE) to mitigate offline dictionary brute-force attacks. WPA3 incorporates the National Security Agency (NSA) Commercial National Security Algorithm Suite (CNSA) set of cryptographic algorithms and also adds Protected Management Frames (PMF) mandatory support for WPA3 certified clients. PMF adds cryptographic protection to deauthentication and dissociation frames, preventing them from being spoofed in a denial-of-service (DoS) attack.

WPA3 Personal uses, at a minimum, the Counter Mode Cipher Block Chaining Message Authentication Code Protocol (CCMP) Advanced Encryption Standard (AES) 128-bit encryption cipher algorithm, and it also supports CCMP-256, as well as Galois/Counter Mode Protocol (GCMP) 128- and 256-bit AES encryption. WPA3 Personal offers both a pure WPA3-SAE mode and a transition mode to allow for WPA2-PSK clients to connect to the SSID.

WPA3 Enterprise supports a 192-bit mode that is achieved by combining 256-bit GCMP-256 AES encryption, 384-bit Hashed Message Authentication Mode (HMAC) with Secure Hash Algorithm (HMAC-SHA384) for key hashing, and Elliptic Curve Diffie-Hellman (ECDH) exchange and Elliptic Curve Digital Signature Algorithm (ECDSA) using a 384-bit elliptic curve for authentication.

Pre-Shared Key Authentication

A key element of authentication is making sure the credentials that the client sends can be read only by the authenticating device and are sent to the correct authenticating device. These credentials are typically encrypted before being sent.

Pre-Shared Key (PSK) authentication uses symmetric encryption, which means the same algorithm and key that are used to encrypt the credentials are used, in reverse, to decrypt the message. With PSK, a common password is configured on both sides. Symmetric key encoding is relatively simple, but it is not recommended for strong user authentication because it is not very resistant to key attacks.

One issue with PSK authentication is that if the password is saved on the wireless client, the process does not authenticate the person who makes the connection; rather, it authenticates the device that makes the connection. The same authentication process occurs whether a device is being used by a valid user or by an attacker. For this reason, storing personal passwords on laptop or desktop computers is considered dangerous. Unless authentication requires the user to enter credentials, the device, not the user, is being authenticated.

Whether the infrastructure authenticates a device or its user, the process occurs when a connection to the network is made. The authentication process begins at either an AP or wireless controller at Layer 2, and authentication can also occur deeper in the network, in which case Layer 3 (IP) is used to communicate between the end device and the authentication server.

PSK Authentication Process

An 802.11-compliant WLAN client uses open authentication by default. Open authentication uses no keys, it operates at Layer 1 and Layer 2, and it does not offer end-to-end security.

With open authentication, the client device only has to authenticate itself as an 802.11-capable device. There is no encryption, per-packet authentication, or message integrity check. Additional security measures should be added, such as IEEE 802.1X.

Shared key authentication is initially like open authentication. However, a key is required on the client and on the AP; by using this key, the authentication process adds a challenge and a response between the AP and the client.

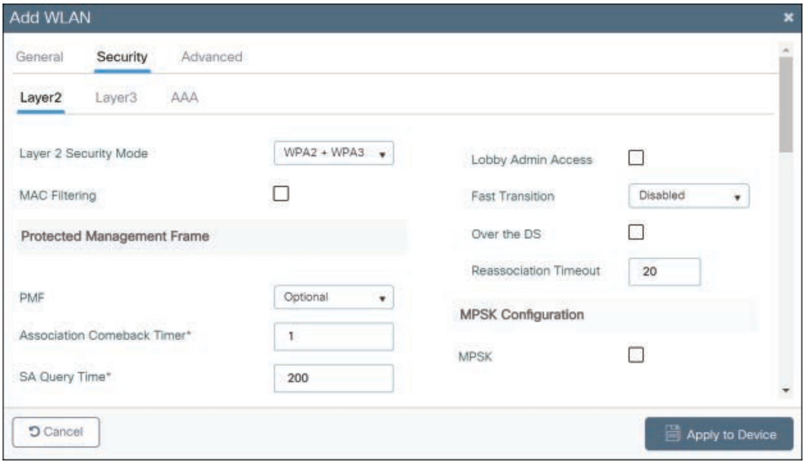
The PSK client authentication and association process are as follows:

1. The client sends an authentication request to the AP.
2. The AP sends a plaintext challenge phrase to the client.
3. The client encrypts the phrase with the shared key and sends it to the AP.
4. If the AP can decrypt the phrase with the key, the AP sends an authentication response to the client.
5. Once authenticated, the client makes an association request.
6. The AP sends an association response.
7. A virtual port is opened, and client data is allowed.
8. Data is encrypted using the same key.

WPA2 and WPA3 PSK Authentication Example

This section provides a configuration example that demonstrates a PSK deployment using a Cisco Catalyst 9800-CL wireless controller. Figure 10-10 through Figure 10-13 show an SSID being configured for WPA2 and WPA3 transition mode.

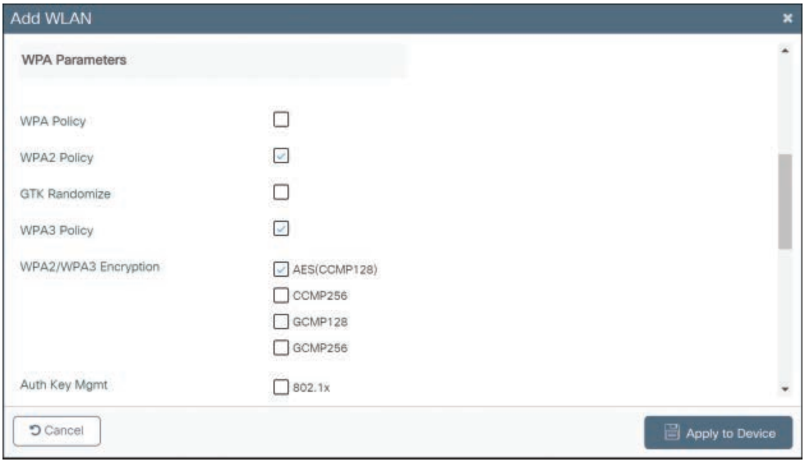
Figure 10-10 Configuring WPA2 and WPA3 PSK Authentication



Layer 2 Security Mode: WPA2 + WPA3
MAC Filtering ☐
Protected Management Frame
PMF: Optional
Association Comeback Timer*: 1
SA Query Time*: 200
Lobby Admin Access ☐
Fast Transition: Disabled
Over the DS ☐
Reassociation Timeout: 20
MPSK Configuration
MPSK ☐

In Figure 10-10, notice that the Layer 2 security mode is set to WPA2 + WPA3 and that PMF has been set to Optional to allow support for WPA2 since PMF is not mandatory under WPA2.

Figure 10-11 Configuring WPA2 and WPA3 Encryption

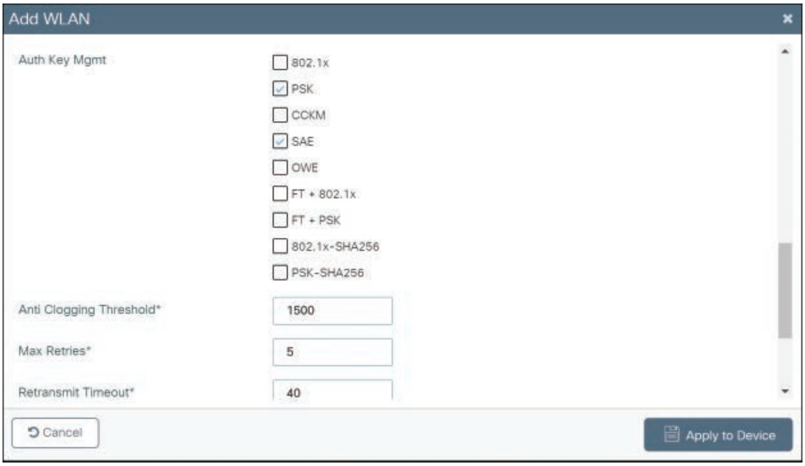


WPA Parameters

WPA Policy ☐
WPA2 Policy ☑
GTK Randomize ☐
WPA3 Policy ☑
WPA2/WPA3 Encryption ☑ AES(CCMP128)
☐ CCMP256
☐ GCMP128
☐ GCMP256
Auth Key Mgmt ☐ 802.1x

Figure 10-11 shows that both WPA2 and WPA3 have been enabled to use AES (CCMP128). Other options include CCMP-256, GCMP-128, and GCMP-256.

Figure 10-12 Configuring WPA2 and WPA3 Authentication Key Management



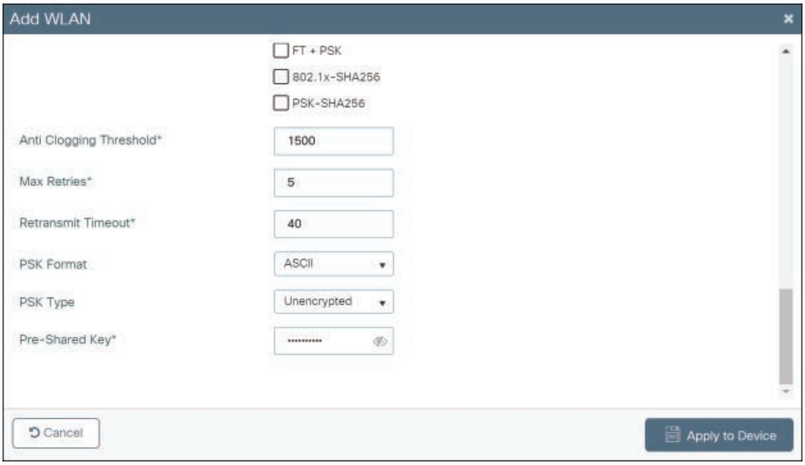
Auth Key Mgmt ☐ 802.1x
☑ PSK
☐ CCKM
☑ SAE
☐ OWE
☐ FT + 802.1x
☐ FT + PSK
☐ 802.1x-SHA256
☐ PSK-SHA256
Anti Clogging Threshold* 1500
Max Retries* 5
Retransmit Timeout* 40

In Figure 10-12, both PSK and SAE are enabled in the Auth Key Mgmt section. These settings allow support of the WPA3 transition mode when WPA2 and WPA3 are both operating on the same SSID.

Finally, in Figure 10-13, a pre-shared key is entered. This key will be used by the clients and the WLC to authenticate users and encrypt/decrypt data.

The only differences between setting up an SSID for WPA2/WPA3 transition mode and setting up a pure WPA3 SSID are that PMF is set to Required, and PSK is disabled, leaving SAE as the only authentication key management mechanism.

Figure 10-13 Configuring WPA2 and WPA3 Pre-Shared Key

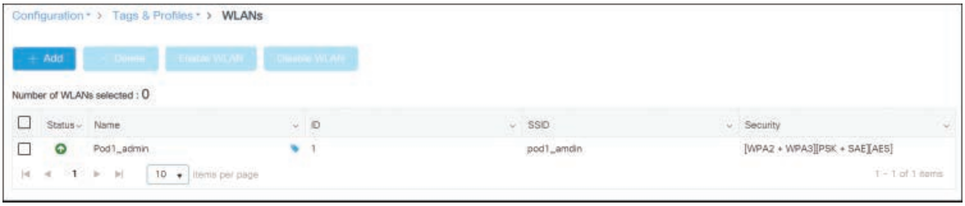


☐ FT + PSK
☐ 802.1x-SHA256
☐ PSK-SHA256

Anti Clogging Threshold*	1500
Max Retries*	5
Retransmit Timeout*	40
PSK Format	ASCII
PSK Type	Unencrypted
Pre-Shared Key*	••••••••••

Figure 10-14 shows the verification of the newly configured WLAN. You can see in this figure that the Pod1_admin WLAN is enabled and has been successfully configured for WPA2 and WPA3 support, using AES.

Figure 10-14 Verifying PSK Authentication



Number of WLANs selected : 0

☐	Status	Name	ID	SSID	Security
☐	●	Pod1_admin	1	pod1_amdin	[WPA2 + WPA3][PSK + SAE][AES]

802.1X User Authentication Overview

Separating authentication from encryption is an important element in improving security. If the authentication method is strong, it is probably too CPU intensive to be used in a process to encrypt each packet. On the other hand, if the algorithm that is used for the encryption process is fast, it is probably too weak to resist a brute-force attack.

A fast algorithm should be used for packet encryption but not to send a password that will remain the same from one session to the next.

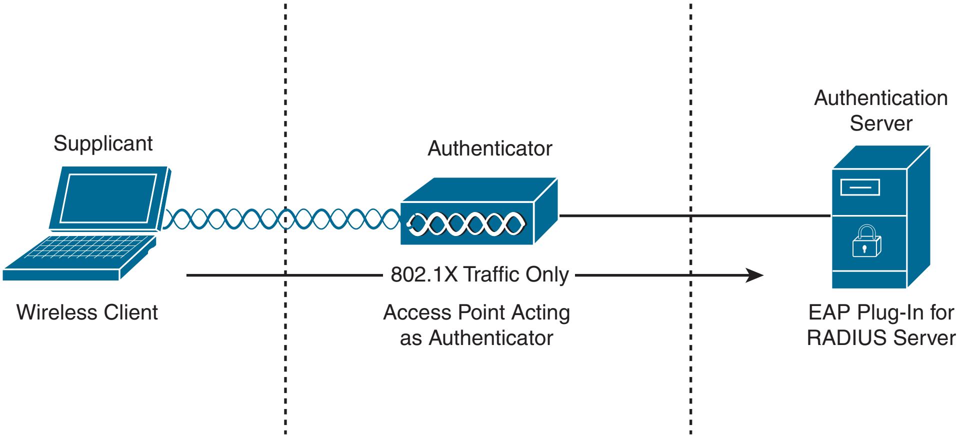
Authentication is the first aspect of wireless security that must be implemented, but a common authentication key for all users in the same WLAN presents too many risks. When one machine key is compromised, all others are exposed. This risk leads to the need for individual authentication keys. However, individual authentication keys lead to issues of scaling the use of too many individual keys.

Fortunately, the same issue has been identified in many other contexts, such as when a user plugs a new device into a switch. Even if physical security limits the number of external users that can access the corporate network, the identity of the user or the nature of the device also needs to be identified before access can be allowed.

The 802.1X protocol defines port-based access control. It defines three roles, as illustrated in Figure 10-15:

- **Supplicant:** The machine (typically a PC or wireless client) that wants to access the network
- **Authenticator:** The point of access (a switch, an AP, or a WLC)—that is, the point of entrance to the network
- **Authentication server:** A machine, somewhere in the network, that keeps a list of conditions according to which access is granted or refused

Figure 10-15 802.1X Process



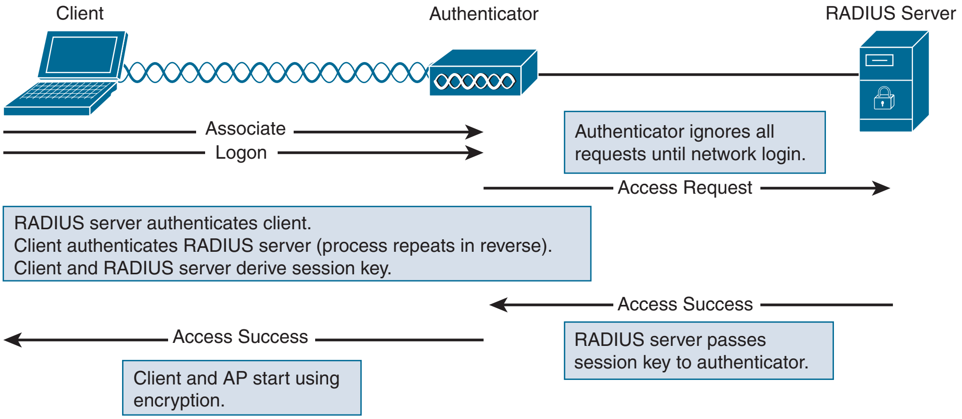
In the 802.1X process, the supplicant connects to the authenticator. At this point, the port on the authenticator is connected from a physical standpoint; however, the 802.1X process has not authorized the port, and no frames are passed from the port on the supplicant to the switching fabric. To be allowed to send and receive traffic, the supplicant needs to send a form of authentication: an ID. If the supplicant that is attached to the network does not send an ID, the port remains unauthorized. In this state, the port cannot pass any user traffic.

The authenticator receives the ID from the client (the supplicant). The authenticator then passes the ID information to an authentication server (typically a RADIUS server) that can verify the identification information. The RADIUS server responds to the authenticator with either a Success or Failure message. If the response is a success, the port is authorized, and user traffic is allowed to pass through the port, as it would pass through any switch port connected to an access device. If the response is a failure, the port remains unauthorized and cannot be used. If there is no response from the server, the port remains unauthorized and does not pass any traffic.

The authenticator can be a switch in a wired network or an AP or a WLC in a wireless network. In the latter case, connecting to the authenticator involves the four steps of an open authentication process, as shown in Figure 10-16:

1. The supplicant sends an authentication request and receives an authentication response with a Success status. For this reason, wireless networks that use 802.1X are often seen by wireless scanners as having open authentication.
2. The supplicant sends an association request and receives an association response with an ID. Even after association is granted, the logical port remains blocked. The wireless client cannot access the network further and must go through the authentication mechanism. The supplicant can start the process, or the authenticator can take the initiative by asking for the credentials. In either case, the supplicant sends authentication credentials to the authenticator.
3. The authenticator receives authentication credentials from the supplicant and encapsulates any 802.1X traffic that is bound for the authentication server and sends it to the server. All other network traffic and all other attempts to access network resources are blocked.
4. After receiving RADIUS traffic that is bound for the client, the AP encapsulates it and sends the information to the client. Although the server authenticates the client as a valid network user, this process allows the client to validate the server as well, ensuring that the client is not logging in to a rogue AP and network.

Figure 10-16 802.1X Client Authentication


Client
Authenticator
RADIUS Server
Associate
Logon
Authenticator ignores all requests until network login.
Access Request
RADIUS server authenticates client.
Client authenticates RADIUS server (process repeats in reverse).
Client and RADIUS server derive session key.
Access Success
RADIUS server passes session key to authenticator.
Access Success
Client and AP start using encryption.

An important aspect of 802.1X is that it authenticates each supplicant individually, thereby improving the authentication phase. Wireless networks usually take advantage of this individual authentication to add an individual encryption layer. While authenticating each other through 802.1X, the client and RADIUS servers derive an individual key, which is unique for the device and the session. Using 802.1X authentication means that each wireless client can be granted a new, dynamic key each time the client accesses the network. Because these keys are dynamic and session based, an intruder cannot learn the system keys and then use them to access the WLAN.

Extensible Authentication Protocol

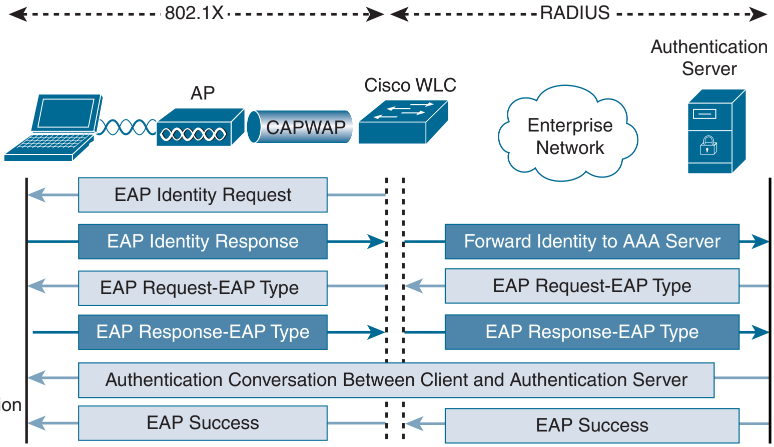
The 802.1X architecture does not include protocol specifics for wireless clients to send their credentials to the authentication server, nor does it include specifies about how this authentication should occur. To solve this problem, the Internet Engineering Task Force (IETF) designed Extensible Authentication Protocol (EAP). EAP is a general protocol for authentication that also supports multiple authentication methods, such as token cards, Kerberos, one-time passwords, certificates, public key authentication, and smart cards.

EAP does not specify which type of authentication to use; it simply defines the authentication steps and headers. EAP separates the authentication itself from the authentication process. You can therefore use EAP with almost any type of authentication, and several layers of consecutive authentications can occur within the same EAP framework.

EAP defines four message types: request, response, success, and failure (see Figure 10-17).

Any side in the authentication process can start, although the AP/WLC usually starts the process by sending an *identity request* message.

Figure 10-17 EAP Process


802.1X
RADIUS
Authentication Server
AP
Cisco WLC
CAPWAP
Enterprise Network
EAP Identity Request
EAP Identity Response
Forward Identity to AAA Server
EAP Request-EAP Type
EAP Request-EAP Type
EAP Response-EAP Type
EAP Response-EAP Type
Authentication Conversation Between Client and Authentication Server
EAP Success
EAP Success

*EAPOL = EAPOL +
Extensible Authentication
Protocol over LAN

Because of this flexibility, several mechanisms are defined to allow client authentication. Some of these mechanisms authenticate the client; some authenticate both the server and the client. Other mechanisms authenticate the device, the user, or both. You can implement a mechanism that suits the level of security needed, as long as that mechanism supports EAP.

There are approximately 40 different types of EAP. Some of the most common types include the following:

- **EAP-TLS (EAP Transport Layer Security):** While very secure, EAP-TLS requires that a client certificate be installed on each Wi-Fi workstation. This approach requires a public key infrastructure (PKI) with extra administrative expertise and time beyond that required for maintaining the WLAN.

- **PEAP (Protected EAP):** PEAP is secure and requires only server-side certificates. Therefore, you can use a more manageable PKI or no PKI. Cisco and Microsoft support PEAP, and it is available at no additional cost from Microsoft.
- **EAP-FAST (EAP Flexible Authentication via Secure Tunneling):** EAP-FAST is a secure solution for enterprises that cannot enforce a strong password policy and do not want to deploy certificates for authentication.
- **EAP-TTLS (EAP Tunneled Transport Layer Security):** EAP-TTLS addresses the certificate issue by tunneling TLS and, thus, eliminating the need for a certificate on the client side. This approach is often the preferred option.

PEAP is currently the most prominently used form of EAP, as it is available on Microsoft servers, but EAP-TLS is gaining in popularity as it can be supported by Cisco ISE.

In wireless networks, all these protocols rely on 802.1X to block data access to the network. These protocols also rely on EAP to carry the authentication exchange between the client, the user or device, and the authentication server. On the wireless client, the type of EAP that is configured must match the configuration on the authentication server.

802.1X and EAP address authentication but not encryption. 802.1X and EAP can be used with or without encryption. For 802.1X and EAP authentication, all packets must be relayed between the client and the authentication server. The content of the EAP messages is of no importance to the controller and AP, which simply relay the information.

The authentication server functionality in the EAP process can be provided either locally or globally:

- **Locally:** EAP can be provided locally by a Cisco wireless LAN controller; this option is referred to as *local EAP*.

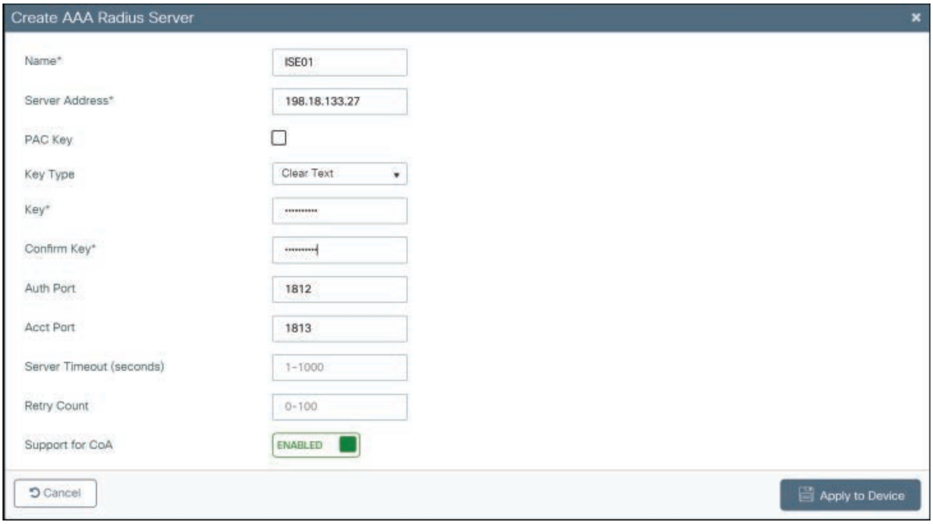
  Local EAP can use either the local user database or a Lightweight Directory Access Protocol (LDAP) database to authenticate users. Local EAP can also be used as a backup for RADIUS authentication. This approach allows wireless clients to authenticate even if the controller loses connectivity to the RADIUS server.
- **Globally:** EAP can be provided globally by a RADIUS server, such as one of these:
  - Cisco Identity Services Engine (ISE)
  - A Microsoft server that is configured for RADIUS
  - Any RADIUS-compliant server

802.1X EAP Authentication Example

This example assumes that a Cisco ISE server is already configured and enabled on the network at address 198.18.133.27 and that the server authentication key is C1sco12345. The first steps in configuring the WLC for 802.1X support is to define (1) a AAA server, (2) a AAA server group, and (3) an authentication method. Optionally, you could also define authorization and accounting methods.

Navigate to Configuration > Security > AAA > Servers / Groups > Servers and click Add to define a new AAA RADIUS server on the WLC, as shown in Figure 10-18. In this example, the server is named ISE01, and it is configured to use the default RADIUS UDP ports. Click Apply to Device to save the server to the WLC.

Figure 10-18 Configuring AAA RADIUS Server Parameters

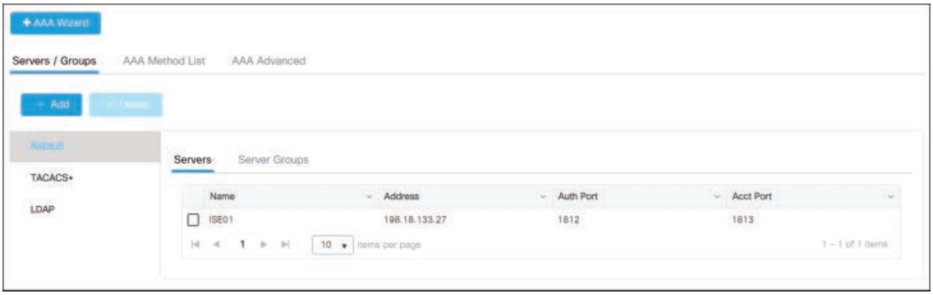

Create AAA Radius Server
×

Name*	ISE01
Server Address*	198.18.133.27
PAC Key	☐
Key Type	Clear Text
Key*	••••••••••
Confirm Key*	••••••••••
Auth Port	1812
Acct Port	1813
Server Timeout (seconds)	1-1000
Retry Count	0-100
Support for CoA	ENABLED

Cancel
Apply to Device

Figure 10-19 shows the AAA Servers/Groups page, where you can confirm that the new AAA server has been successfully saved.

Figure 10-19 Verifying AAA Server

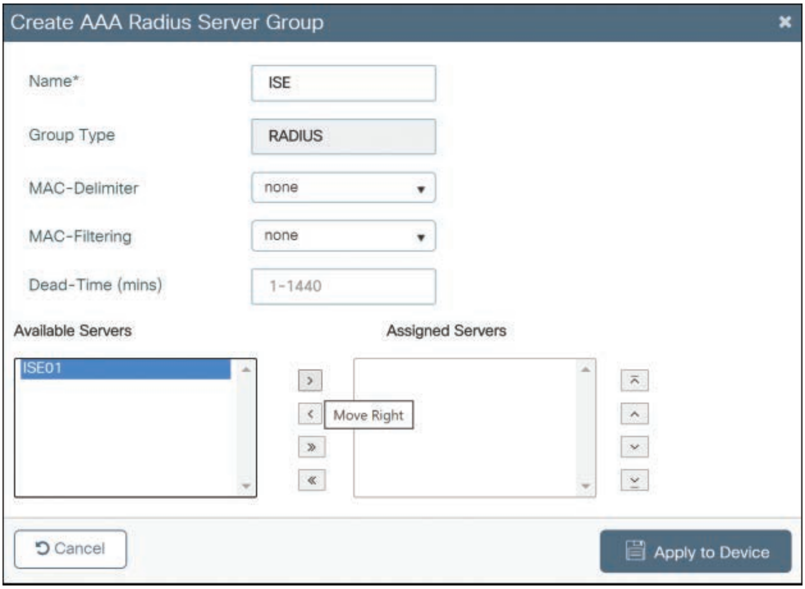

+ AAA Wizard
Servers / Groups
AAA Method List
AAA Advanced
+ Add
Delete
RADIUS
TACACS+
LDAP
Servers
Server Groups

	Name	Address	Auth Port	Acct Port
☐	ISE01	198.18.133.27	1812	1813

1
10
items per page
1 - 1 of 1 items

The next step is to define a AAA server group to include the newly configured AAA server. Click Server Groups in the subtab and then click Add. Figure 10-20 shows that a AAA server group has been named ISE. Notice that the ISE01 server appears in the Available Servers list. By clicking the right arrow icon, you can move the server to the Assigned Servers list. Click Apply to Device to save the server group to the WLC.

Figure 10-20 Configuring AAA Server Group


Create AAA Radius Server Group

Name* ISE

Group Type RADIUS

MAC-Delimiter none

MAC-Filtering none

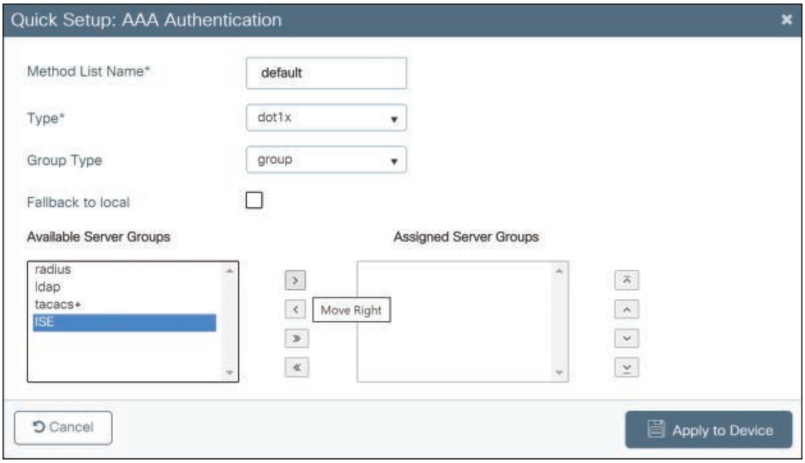
Dead-Time (mins) 1-1440

Available Servers: ISE01

Assigned Servers:

Next, you need to define a AAA authentication method. Navigate to Configuration > Security > AAA > AAA Method List > Authentication and then click Add. In Figure 10-21, the default method list name is used. The type of authentication is changed to 802.1X (dot1x), and the group type is left to group because this example uses an external server instead of local authentication. Finally, the ISE server is moved from the Available Server Groups list to the Assigned Server Groups list. Click Apply to Device to save the configuration to the WLC.

Figure 10-21 Configuring AAA Authentication


Quick Setup: AAA Authentication

Method List Name* default

Type* dot1x

Group Type group

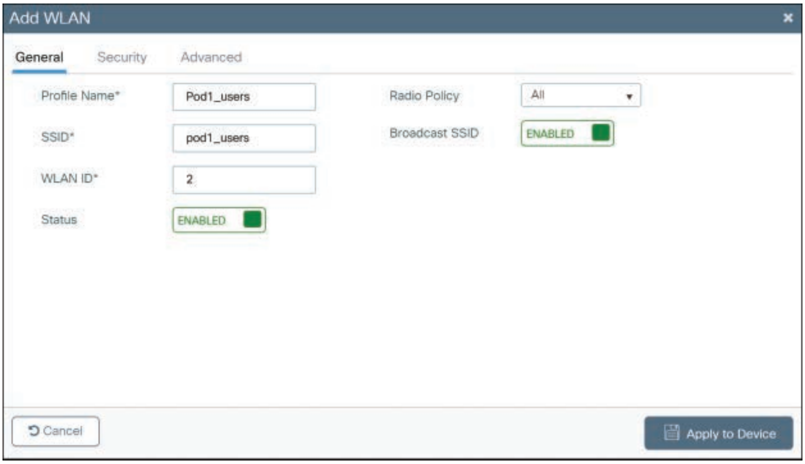
Fallback to local ☐

Available Server Groups: radius, ldap, tacacs+, ISE

Assigned Server Groups:

Once the AAA server is configured on the WLC, a WLAN needs to be created that will use 802.1X for its authentication method. From the WLC, Navigate to Configuration > Tags and Profiles > WLANs. Click Add to create a new WLAN. In Figure 10-22, a WLAN called Pod1_users is defined and enabled under the General tab.

Figure 10-22 Configuring a New WLAN

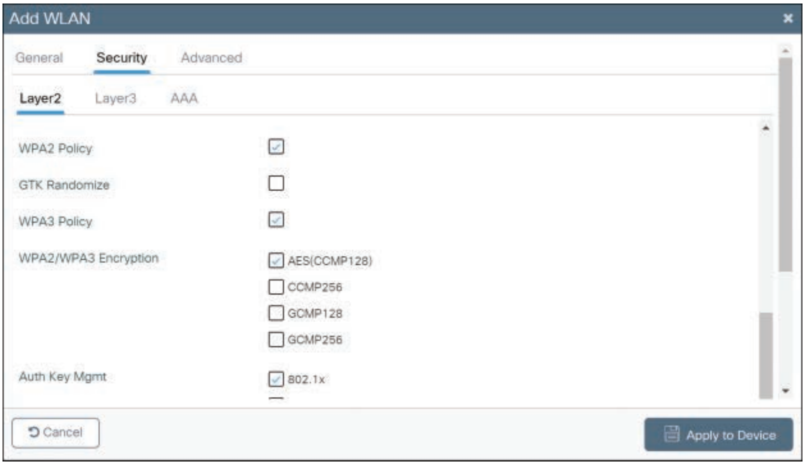

Add WLAN
×
General
Security
Advanced
Cancel
Apply to Device

Profile Name* Pod1_users
SSID* pod1_users
WLAN ID* 2
Status ENABLED
Radio Policy All
Broadcast SSID ENABLED

If you click on the Security tab, you can set up Layer 2 parameters, as shown in Figure 10-23. Here, WPA2 and WPA3 are enabled, using AES (CCMP128) for encryption and 802.1X for authentication.

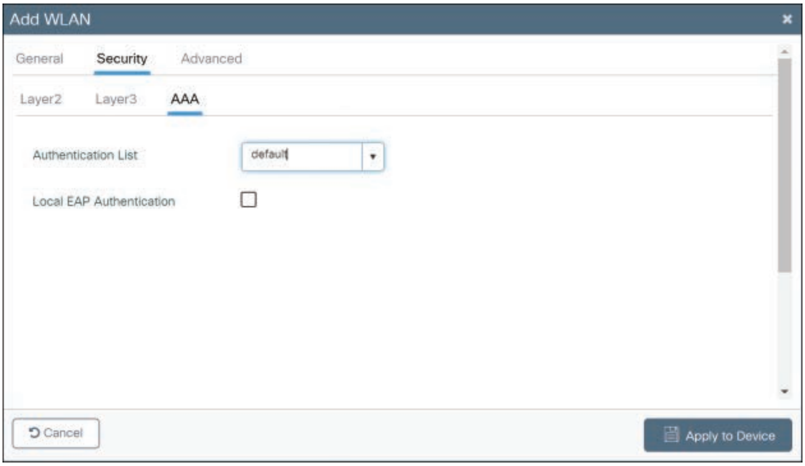
Next, click on the AAA tab and select default from the Authentication List dropdown menu, as shown in Figure 10-24. Recall that this is the AAA authentication method defined earlier. Click Apply to Device to save the configuration to the WLC.

Figure 10-23 Configuring 802.1X Authentication for WPA2 and WPA3


Add WLAN
×
General
Security
Advanced
Layer2
Layer3
AAA
Cancel
Apply to Device

WPA2 Policy ☑
GTK Randomize ☐
WPA3 Policy ☑
WPA2/WPA3 Encryption ☑ AES(CCMP128)
☐ CCMP256
☐ GCMP128
☐ GCMP256
Auth Key Mgmt ☑ 802.1x

Figure 10-24 Configuring the AAA Authentication Method for the WLAN

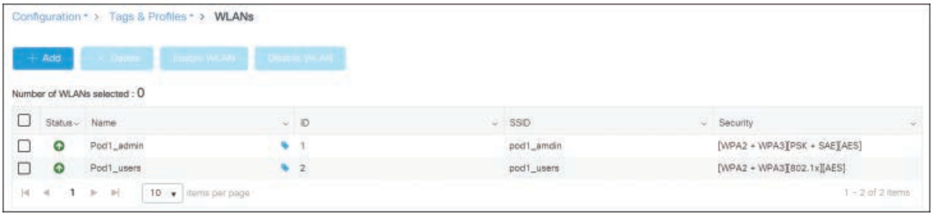

Add WLAN
×
General
Security
Advanced
Layer2
Layer3
AAA
Cancel
Apply to Device

Authentication List: default

Local EAP Authentication ☐

Figure 10-25 shows that the WLC now has two WLANs configured. The first, Pod1_admin, was defined earlier using PSK authentication. The second, Pod1_users, is configured for WPA2 and WPA3 using AES for encryption and 802.1X for authentication.

Figure 10-25 Verifying the 802.1X WLAN Configuration


Configuration > Tags & Profiles > WLANs
+ Add
Delete
Enable WLAN
Disable WLAN
items per page
1 - 2 of 2 items

Number of WLANs selected : 0

☐	Status	Name	ID	SSID	Security
☐	●	Pod1_admin	1	pod1_amdin	[WPA2 + WPA3][PSK + SAE][AES]
☐	●	Pod1_users	2	pod1_users	[WPA2 + WPA3][802.1x][AES]

Guest Access with Web Auth

One of the challenges in a wireless environment is providing guest access. Guests must be able to easily access the wireless network, yet they must not compromise the security of the corporate network. Another factor is balancing security with administrative ease.

Web Authentication (WebAuth) is a process that allows users, typically guests, to authenticate to a network through a web portal via a browser interface. Clients that attempt to access the WLAN using HTTP are automatically redirected to a login page, where they are prompted for their credentials. Their credentials are then passed to an authentication server, which then assigns the appropriate VLAN and security profile for guest access to the Internet.

Three basic areas must be defined for web authentication:

- From where guest path isolation is defined in the network**:**
    - Local WLC
    - Auto-Anchor

- From where web portal pages are provisioned:
  - Local pages on a WLC
  - Remote pages on an external web server
- From where users are defined:
  - Local guest user account on a WLC
  - Centralized guest user account on a RADIUS authentication server

Local Web Authentication

Local WebAuth is designed for small businesses that need to provide local guest access. The VLAN that is used is defined in the network and routed to the Internet via the router and firewall at the network edge.

The WLC has a default web login page that can be used for guest authentication. You can use the default page as is or make minor modifications such as hiding the Cisco logo, creating your own headline for the login page, and redirecting the user to another URL via HTTP after login.

You can also download a customized WebAuth login page to the WLC. The customized page can have additional elements such as an acceptable use policy (AUP) that must be confirmed with the guest user login.

A lobby ambassador management account can be created for each WLC to offload guest access provisioning from the IT staff. This account is a management user account with the specific role of creating, deleting, and extending the use of guest accounts.

Local Web Authentication with Auto-Anchor

Some customers prefer to have all guest traffic on a single network, but the dedicated guest VLAN and dedicated port concepts do not scale well across the enterprise. These issues can be addressed by using guest anchors.

Auto-anchor mobility (also called *guest tunneling*) is a feature of mobility to restrict a WLAN to a single subnet, regardless of a client entry point into the network.

Auto-anchor operates as follows:

1. The guest associates to the local controller, and a local session is created.
2. A session (via a tunnel) is created to the Auto-Anchor WLC. (The session is per SSID, not per client.)
3. Packets from the client are encapsulated and sent through the tunnel to the Auto-Anchor WLC.
4. The Auto-Anchor WLC decapsulates the client packets and delivers them to the wired network.
5. Traffic from the wired network to the client goes through the same tunnel.

Local Web Portal with External Authentication

Rather than have the WebAuth process be handled exclusively by the local WLC, the authentication portion can be separated and handled by an external authentication server, while the local WLC handles the provisioning of the web login pages. This model works much like a local WLC and authentication server, but an auto-anchor WLC can optionally be used for path isolation.

Local WebAuth with an external authentication operation works as follows:

1. A guest associates to a local controller, and a local session is created.
2. The guest receives web login pages from the local WLC.
3. The guest enters credentials that are forwarded to an authentication server (Cisco ISE) for authentication.
4. The authentication server returns confirmation (assuming that the credentials are valid).
5. Guest traffic is routed to the Internet, and the WLC provides path isolation (with or without the use of auto-anchor).

Centralized Web Authentication

With Centralized WebAuth, the tasks of provisioning web login pages and maintaining the guest user accounts are not done on WLCs but are instead done by a centralized server, such as Cisco ISE.

Cisco ISE is more than just a RADIUS server. Cisco ISE allows you to provide highly secure network access to users and devices. It helps you gain visibility into what is happening in your network, such as who is connected, which applications are installed and running, and much more. It also shares vital contextual data, such as user and device identities, threats, and vulnerabilities, with integrated solutions from Cisco technology partners, so you can identify, contain, and remediate threats more quickly.

Centralized WebAuth operates as follows:

1. A guest associates to a local controller, and a local session is created.
2. The guest is redirected to Cisco ISE.
3. Cisco ISE provides web portal pages and guest authentication.
4. Guest traffic is routed to the Internet.

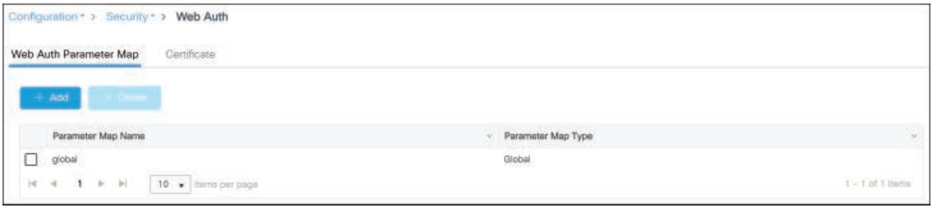
Web Auth Authentication Configuration Example

This example shows how to configure a Cisco Catalyst 9800 wireless controller to create a Guest WLAN that uses Web Auth authentication.

Although it is possible to serve web pages with an extensive AUP from a WLC or from an external web server before the user connects to the network, in this example the user receives a basic policy page that is sourced from the WLC.

The first step is to navigate to Configuration > Security > Web Auth. Figure 10-26 shows that the WLC currently has one parameter map, called global. Clicking this default predefined parameter map allows you to edit it.

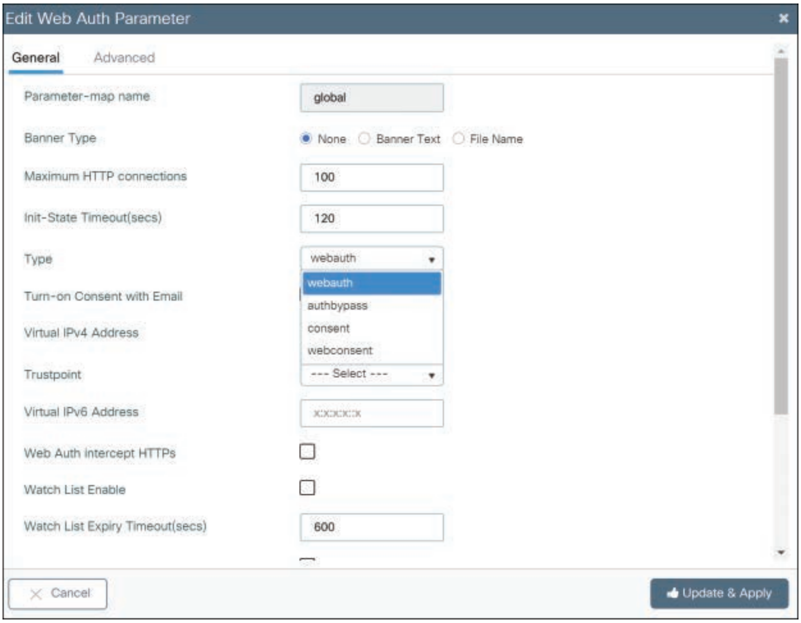
Figure 10-26 Accessing Web Auth Configuration Window


Configuration > Security > Web Auth
Web Auth Parameter Map
Certificate
+ Add
× Delete
10
items per page
1 - 1 of 1 items

Parameter Map Name	Parameter Map Type
☐ global	Global

In the Edit WebAuth Parameter window that appears, click the Type dropdown menu, as shown in Figure 10-27.

Figure 10-27 Editing WebAuth Parameters


Edit Web Auth Parameter
General
Advanced
webauth
authbypass
consent
webconsent
Cancel
Update & Apply

Parameter-map name	global
Banner Type	● None ○ Banner Text ○ File Name
Maximum HTTP connections	100
Init-State Timeout(secs)	120
Type	webauth
Turn-on Consent with Email	
Virtual IPv4 Address	
Trustpoint	--- Select ---
Virtual IPv6 Address	X:X:X:X::X
Web Auth intercept HTTPs	☐
Watch List Enable	☐
Watch List Expiry Timeout(secs)	600

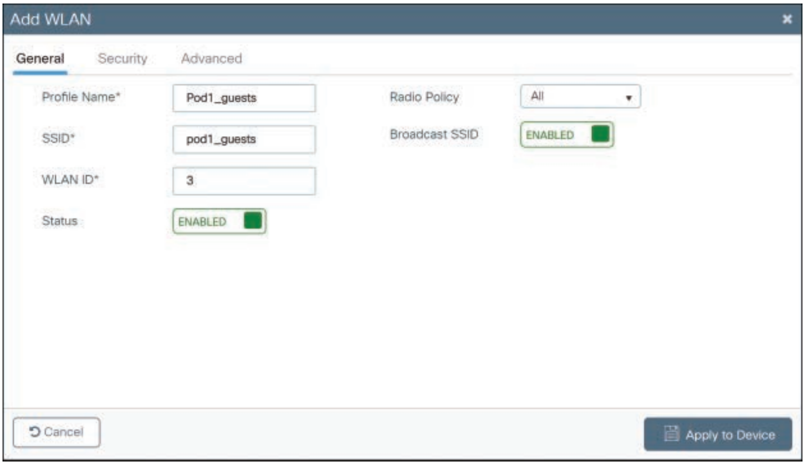
Notice the WebAuth option types that are available:

- **webauth:** This option is a basic WebAuth. The controller presents a policy page with the username and password, and the user needs to enter the correct credentials to access the network.
- **authbypass and consent:** With this option, the controller presents a policy page with the Accept or Deny buttons, and the user needs to click the Accept button to access the network.
- **webconsent:** This option is a combination of the webauth and consent web authentication types just described. The controller presents a policy page with Accept or Deny buttons along with username or password, and the user needs to enter the correct credentials and click the Accept button to access the network.

Notice the Banner Type option, where you can optionally either specify text or reference a file that contains text that will be displayed on the policy page when it is presented to the user.

Select webauth and click Update & Apply. Next, navigate to Configuration > Tags & Profiles > WLANs and then click Add. A new WLAN is defined, called Pod1_guests, as shown in Figure 10-28.

Figure 10-28 Defining a New WLAN for Guest Access


Add WLAN
×
General
Security
Advanced
Cancel
Apply to Device

Profile Name* Pod1_guests

SSID* pod1_guests

WLAN ID* 3

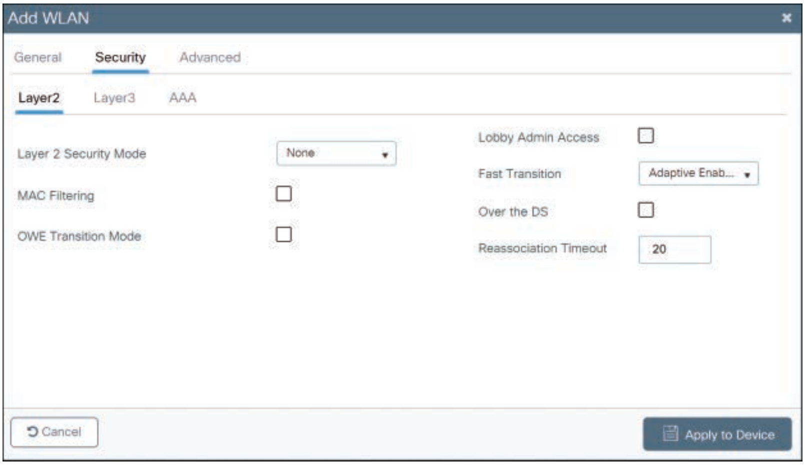
Status ENABLED

Radio Policy All

Broadcast SSID ENABLED

Click the Security tab. Under the Layer 2 subtab, select None from the Layer 2 Security Mode dropdown menu, as shown in Figure 10-29.

Figure 10-29 Defining Layer 2 Security for Guest Access


Add WLAN
×
General
Security
Advanced
Layer2
Layer3
AAA
Cancel
Apply to Device

Layer 2 Security Mode None

MAC Filtering ☐

OWE Transition Mode ☐

Lobby Admin Access ☐

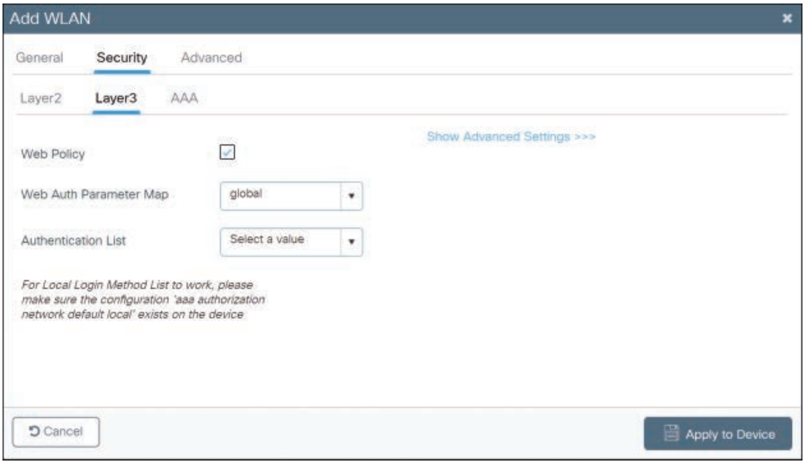
Fast Transition Adaptive Enab...

Over the DS ☐

Reassociation Timeout 20

Under the Layer 3 subtab, check the Web Policy box and select the global from the WebAuth Parameter Map dropdown menu, as shown in Figure 10-30. This enables guest authentication for the Pod1_guests WLAN.

Figure 10-30 Defining Layer 3 Security for Guest Access


Add WLAN
General
Security
Advanced
Layer2
Layer3
AAA
Show Advanced Settings >>>
Cancel
Apply to Device

Web Policy ☑

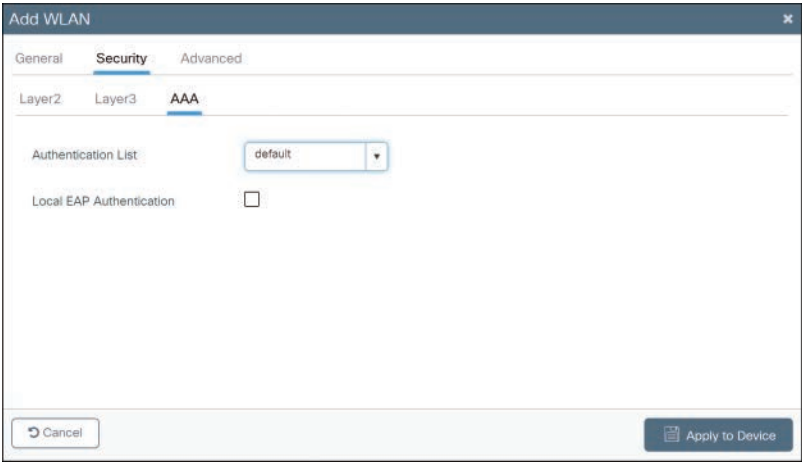
Web Auth Parameter Map: global

Authentication List: Select a value

For Local Login Method List to work, please make sure the configuration 'aaa authorization network default local' exists on the device

The final step is to enable AAA authentication under the AAA subtab, as shown in Figure 10-31. The only value available under the Authentication List dropdown menu is the default authentication method defined earlier on the WLC. This allows authentication to occur with the ISE RADIUS server.

Figure 10-31 Defining AAA Authentication for Guest Access


Add WLAN
General
Security
Advanced
Layer2
Layer3
AAA
Cancel
Apply to Device

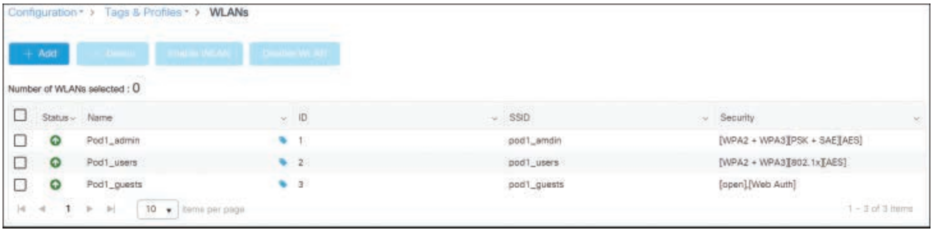
Authentication List: default

Local EAP Authentication ☐

Click Apply to Device to save the configuration to the WLC.

The WLC now has a third WLAN, called Pod1_guests, and it is configured for open access but uses WebAuth for authentication, as illustrated in Figure 10-32.

Figure 10-32 Verifying WebAuth Guest Access



Configuration > Tags & Profiles > WLANs
+ Add

Number of WLANs selected : 0

☐	Status	Name	ID	SSID	Security
☐	●	Pod1_admin	1	pod1_amdin	[WPA2 + WPA3][PSK + SAE][AES]
☐	●	Pod1_users	2	pod1_users	[WPA2 + WPA3][802.1x][AES]
☐	●	Pod1_guests	3	pod1_guests	[open],[Web Auth]

1
10 items per page
1 - 3 of 3 items

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	19, 20
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	12

This page intentionally left blank

Day 9

Secure Network Access

ENCOR 350-401 Exam Topics

Security

- Describe the components of network security design
  - Threat defense
  - Endpoint security
  - Next-generation firewall
  - TrustSec, MACsec
  - Network access control with 802.1X, MAB, and WebAuth

Key Topics

Today we start our review of network security concepts by focusing on the enterprise network security architecture. Today's enterprise networks carry sensitive data belonging to both the enterprise and its customers. Understanding threats that face networks and ways to address these threats is an important part of operations in today's enterprise networks. Today we also look at how we can apply a layered approach to defending the network by using different security tools and services.

Network Security Threatscape

Threats to the enterprise network can come in many forms and from many sources, both external and internal. This multitude of threats and sources is referred to as the *threatscape*.

It is easy to recognize the need for improved network security just by monitoring the news. Every few months, there is a new story about large companies falling victim to attacks, and thereby losing huge amounts of private data and intellectual property. No industry is exempt. Companies in the financial, retail, entertainment, energy, and technology industries have all been compromised.

Attackers are not limited to individuals or small teams of hackers. Organized crime and even national governments (state-sponsored cyberattacks) are often implicated in attacks. Today's attackers are exceedingly clever and devious and have vast supporting funding and resources.

As a defender, you must not be restricted by preconceptions of how things are designed to work or strict classification of known network threats. Attackers don't restrict themselves. Attackers are creative thinkers. Attackers combine old and new techniques to produce unique new threats. As a defender, you too must be prepared to think outside the box and evolve to respond to the ever-changing threatscape.

Here is some terminology that will help you as you learn about today's threatscape:

- **Vulnerability:** A weakness that compromises either the security or the functionality of a system. Weak or easily guessed passwords are considered vulnerabilities.
- **Exploit:** The mechanism that is used to leverage a vulnerability to compromise the security or functionality of a system. An example of an exploit is an exploit tool. When a vulnerability is disclosed to the public, attackers often create a tool that implements an exploit for that specific vulnerability. If they release the tool to the Internet, other attackers with very little skill can effectively exploit the vulnerability.
- **Threat:** Any circumstance or event that has the potential to cause harm to an asset in the form of destruction, disclosure, adverse modification of data, or DoS. An example of a threat is malicious software that targets workstations.
- **Risk:** The likelihood that a particular threat using a specific attack will exploit a particular vulnerability of an asset and result in an undesirable consequence.

The threatscape is a very large concept that is constantly growing. Many threat vectors can impact the enterprise network, from many different types of attackers. To help understand the range of threats, here is a summary of some of the major types of attacks and threats:

- **DoS and DDoS:** Denial-of-service (DoS) and distributed denial-of-service (DDoS) attacks attempt to consume all of a critical computer or network resource to make it unavailable for valid use. A DoS attack is initiated by a single source, whereas a DDoS attack comes from a multitude of sources.
- **Spoofing:** An attack can be considered a spoofing attack any time an attacker injects traffic that appears to be sourced from a trusted system other than the attacker's system itself. Spoofing is not specifically an attack, but spoofing can be incorporated into various types of attacks.
- **Reflection:** A reflection attack is a type of DoS attack in which the attacker sends a flood of protocol request packets to various IP hosts. The attacker spoofs the source IP address of the packets such that each packet has as its source address the IP address of the intended target rather than the IP address of the attacker. The IP hosts that receive these packets become "reflectors." The reflectors respond by sending response packets to the spoofed address (the target), thus flooding the unsuspecting target.
- **Social engineering:** Social engineering involves manipulating people and capitalizing on expected behaviors. Social engineering often involves utilizing social skills, relationships, or understanding of cultural norms to manipulate people inside a network to provide the information that is needed to access the network.
- **Phishing:** Phishing is a common social engineering technique. Typically, a phishing email pretends to be from a large, legitimate organization. The large organization is legitimate, and

the target may have a real account with the organization. The malicious website generally resembles that of the real organization. The goal is to get the victim to enter personal information such as account numbers, Social Security numbers, usernames, or passwords. Phishing can also be conducted using the telephone system to achieve the same kind of social engineering. In that case, it is called *vishing*.

- **Password attacks:** Password attacks have been a problem since network security came into being, and they continue to be a dominant problem in current network security. An attacker tries to access protected resources by obtaining a user's password. Methods for retrieving the user's password include guessing, brute-force, and dictionary attacks. Many authentication systems require a certain degree of password complexity. Specifying a minimum length for a password and forcing an enlarged character set (uppercase, lowercase, numeric, and special characters) can have an enormous influence on the feasibility of brute-force attacks.
- **Reconnaissance attacks:** A reconnaissance attack is an attempt to learn more about the intended victim before mounting a more intrusive attack. Attackers can use standard networking tools such as the command-line tools **dig**, **nslookup**, and **whois** to gather public information about a target network from DNS registries. The **nslookup** and **whois** tools are available on Windows, UNIX, and Linux platforms, and **dig** is available on UNIX and Linux systems.
- **Buffer overflow attacks:** Attackers can analyze network server applications for flaws. A buffer overflow vulnerability is one type of flaw. If a service accepts input and expects the input to be within a certain size but does not verify the size of input upon reception, it may be vulnerable to a buffer overflow attack. This means that an attacker can provide input that is larger than expected, and the service will accept the input and write it to memory, filling up the associated buffer and overwriting adjacent memory. This overwrite may corrupt the system and cause it to crash, resulting in DoS. In the worst cases, the attacker can inject malicious code in the buffer overflow, which may lead to a system compromise.
- **Man-in-the-middle attacks:** Generally, in a man-in-the-middle attack, a system that has the ability to view the communication between two systems imposes itself in the communication path between those other systems. Man-in-the-middle attacks are complex and require successful attacks against protocols such as Address Resolution Protocol (ARP), Domain Name System (DNS), or Dynamic Host Configuration Protocol (DHCP), resulting in the misdirection of traffic.
- **Malware:** Malware is malicious software that comes in several forms, including viruses, worms, and Trojan horses. The common thread of these attacks is that the attacker tries to install software on the victim's system. Once the software is installed on the victim's system, the attacker can take control of that system, encrypt and lock the victim's data, or escalate privileges to other parts of the victim's network as part of an advanced persistent threat (APT).
- **Vectors of data loss and exfiltration:** The expression *vector of data loss and exfiltration* refers to the means by which data leaves an organization without authorization. While not a direct attack itself, it is a major security concern in the enterprise network. Many of the tools that make our jobs easier today also make it possible for confidential data to be obtained by unauthorized persons. Some common data loss vectors include email attachments, unencrypted devices, cloud storage services, removable storage devices, and improper access controls.

- **Hacking tools:** The distinction between a security tool and an attack tool is in the intent of the user. A penetration tester legitimately uses tools to attempt to penetrate an organization's security defenses, and the organization uses the results of the penetration test to improve its security defenses. However, an attacker can also use the same tools that the penetration tester uses. Hacking tools include sectools.org, Kali Linux, and Metasploit.

For more information about threats facing today's enterprise network, visit Cisco Talos, at www.talosintelligence.com.

Network Security Components

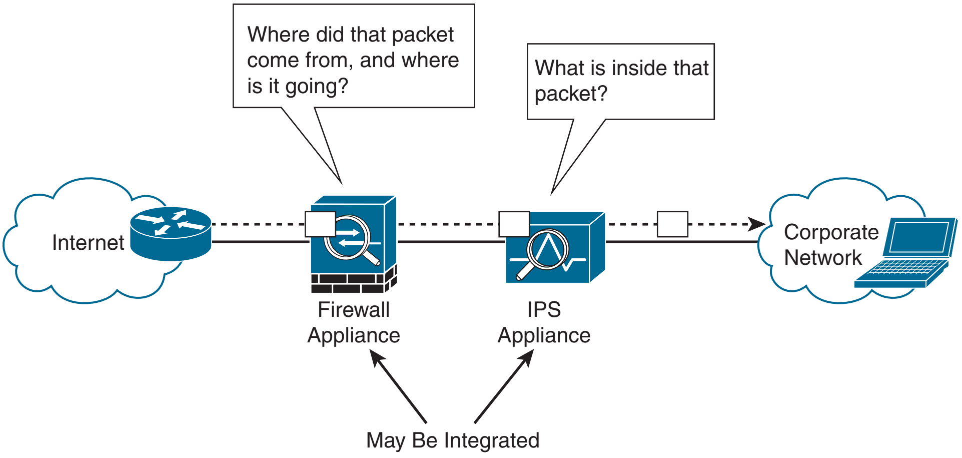
This section describes different threat defense solutions and components that can be deployed within a network to protect it against the attacks listed earlier.

Intrusion Prevention Systems

An intrusion prevention system (IPS) is a system that performs deep analysis of network traffic, searching for signs of suspicious or malicious behavior. If it detects such behavior, the IPS can take protective action. Because it can perform deep packet analysis, an IPS can complement a firewall by blocking attacks that would normally pass through a traditional firewall device. For example, an IPS can detect and block a wide range of malicious files and behavior, including botnet attacks, malware, and application abuse.

Figure 9-1 shows a firewall and an IPS working in conjunction to defend a network. This is an example of network-based IPS, in which IPS devices are deployed at designated network points to address network attacks, regardless of the location of the attack target. Network-based IPS technology is deployed in a sensor, which can be a dedicated appliance or a module that is installed in another network device. There are also host-based IPSs that only detect attacks that occur on the hosts on which they are installed.

Figure 9-1 Intrusion Prevention System


Where did that packet come from, and where is it going?
What is inside that packet?
Internet
Firewall Appliance
IPS Appliance
Corporate Network
May Be Integrated

IPSs use several methods of traffic inspection:

- **Signature-based inspection:** A signature-based IPS examines the packet headers or data payloads in network traffic and compares the data against a database of known attack signatures. The database must be continually updated to remain effective. A signature might be a sequence or a string of bytes in a certain context. Signature-based inspection is sometimes referred to as rule-based or pattern-matching inspection.
- **Anomaly-based inspection:** Anomaly-based network IPS devices observe network traffic and act if a network event outside normal network behavior is detected.

There are three types of anomaly-based network inspection:

- **Statistical anomaly detection (network behavior analysis):** This type of inspection involves observing network traffic over time and building a statistical profile of normal traffic behavior, based on communication patterns, traffic rate, mixture of protocols, and traffic volume. After a normal profile has been established, statistical anomaly detection systems detect or prevent activity that violates the normal profile.
- **Protocol verification:** This type of inspection involves observing network traffic and comparing network, transport, and application layer protocols that are used inside network traffic to protocol standards. If a deviation from standards-based protocol behavior (such as a malformed IP packet) is detected, the system can take appropriate action.
- **Policy-based inspection:** A policy-based IPS analyzes network traffic and takes action if it detects a network event outside configured traffic policy.

Modern IPSs, which are often called next-generation IPSs (NGIPSs), combine the benefits of these inspection methods. They utilize technology such as traffic normalization and protocol decoding to counter evasive attacker techniques and to improve efficacy. They also utilize newer and more sophisticated technologies, such as reputation, context awareness, event correlation, and cloud-based services, to provide more robust and flexible protection. IPS services are often combined with firewall appliances or services, as is the case with next-generation firewalls (NGFW).

Virtual Private Networks

As discussed on Day 20, "GRE and IPsec," a virtual private network (VPN) is a technology that secures communication across an untrusted network. As defined in RFC 2828 and RFC 4949, a VPN is "a restricted-use, logical (i.e., artificial or simulated) computer network that is constructed from the system resources of a relatively public, physical (i.e., real) network (such as the Internet), often by using encryption (located at hosts or gateways), and often by tunneling links of the virtual network across the real network."

A VPN carries private traffic over a public or shared infrastructure (such as the Internet). The most common and effective VPN technology is applied at the network layer of the OSI model to encrypt traffic flow among specific users, applications, or IP subnet pairs. A VPN at the network layer is transparent to the applications at higher OSI layers and is also independent of network topology.

VPNs are classified according to the following criteria:

- **Deployment mode:** A VPN may be deployed as a site-to-site VPN or as a remote-access VPN. A site-to-site VPN provides an Internet-based WAN infrastructure for connecting branch offices, home offices, or the sites of business partners to all or portions of a network. A remote-access VPN provides secure communications for remote access to networks and applications. Hosts can establish remote-access VPNs either by using VPN client software or by using an SSL-enabled web browser.

- **Underlying technology:** VPNs can be classified according to the technology they use, including IPsec VPN, SSL VPN, Multiprotocol Label Switching (MPLS) VPN, other Layer 2 technologies such as Frame Relay or Asynchronous Transfer Mode (ATM), and hybrid VPNs combining multiple technologies.

Content Security

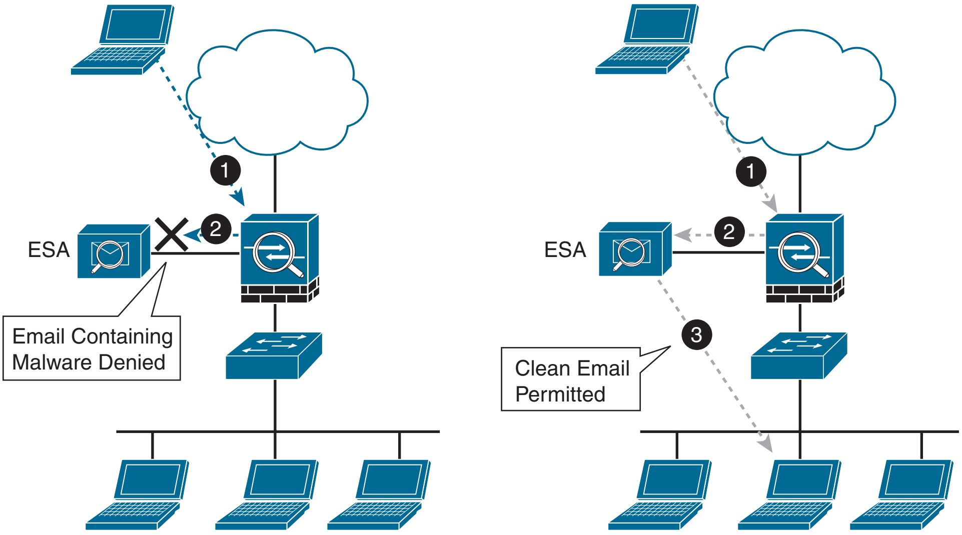
Content security systems provide fine-grained control and security for particular network applications. Examples of Cisco content security products include Cisco Email Security Appliance (ESA) and Cisco Web Security Appliance (WSA).

The Cisco ESA is a type of firewall and threat monitoring appliance for SMTP traffic that offers the following:

- The capability to quickly block new email-based blended attacks
- The capability to control or encrypt sensitive outbound email
- A rapid spam capture rate and few false positives
- A proven zero-hour antivirus solution

The left side of Figure 9-2 shows that when an inbound email arrives (step 1), the Cisco Adaptive Security Appliance (ASA) intercepts that email and forwards it to the ESA (step 2), where the malware is denied. The right of the figure shows that an email that does not contain malware is permitted to reach the intended receiver (step 3).

Figure 9-2 Cisco Email Security Appliance Filtering

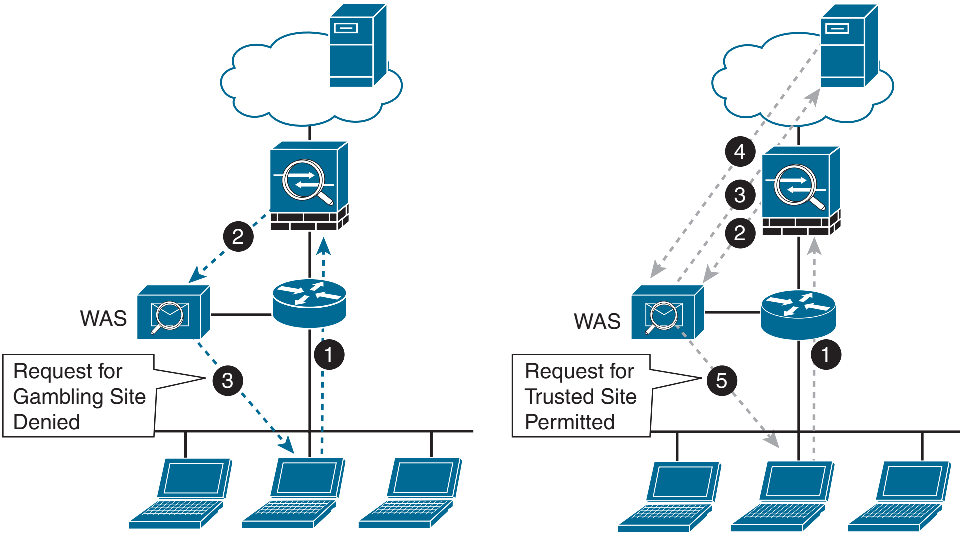


The Cisco WSA provides secure web access, content security, and threat mitigation for web services. In addition to helping to secure and control web traffic, the Cisco WSA provides the following forms of protection:

- Advanced malware protection
- Application visibility and control
- Insightful reporting
- Secure mobility

The left side of Figure 9-3 shows a user's request for a gambling website (step 1) being forwarded to the WSA for inspection (step 2). The WSA blocks the user request and returns a web page informing the user of the policy violation (step 3).

Figure 9-3 Cisco Web Security Appliance Filtering



The right side of the figure shows a user's request for a permitted website (step 1) being forwarded to the WSA (step 2). The WSA proxies the user's request to the website (step 3), and the web page is returned to the WSA (step 4). The WSA can then check the content for malware and forward the verified content to the user (step 5).

Endpoint Security

The term *endpoint security* refers to protecting an endpoint device, such as a desktop computer, laptop, tablet, or smartphone. The term also refers to verifying the user, the device, and the device state to protect the network.

Traditionally, endpoint security has relied on three types of software installed on endpoints to protect endpoint devices and the data that resides on them:

- **Personal firewalls:** This software protects only the device on which it is installed. A personal firewall may have the ability to block applications, files, and services, and it may also provide intrusion detection services.
- **Antivirus software:** This software prevents and removes computer viruses and other types of malicious software. Computer viruses spread from one computer to another, leaving infections as they travel. They can damage data or software or even cause DoS conditions.
- **Antispyware software:** This software detects and removes spyware. Spyware displays advertisements and tracks information on an endpoint device without the user's consent. Spyware can make changes to the endpoint device without the user's consent and can damage the device. Both antispyware and antivirus software must be updated frequently to remain effective.

Today's attackers have the resources, expertise, and persistence to compromise any organization if given enough time. Traditional personal firewalls, antivirus, and antispyware no longer work against these attackers. Defending against today's targeted, persistent malware attacks is a bigger problem than a single point-in-time control or product can effectively address on its own. Therefore, the process of handling malware has evolved. Advanced malware protection uses integrated controls and a continuous process to detect, confirm, track, analyze, and remediate threats before, during, and after an attack.

Centralized Endpoint Policy Enforcement

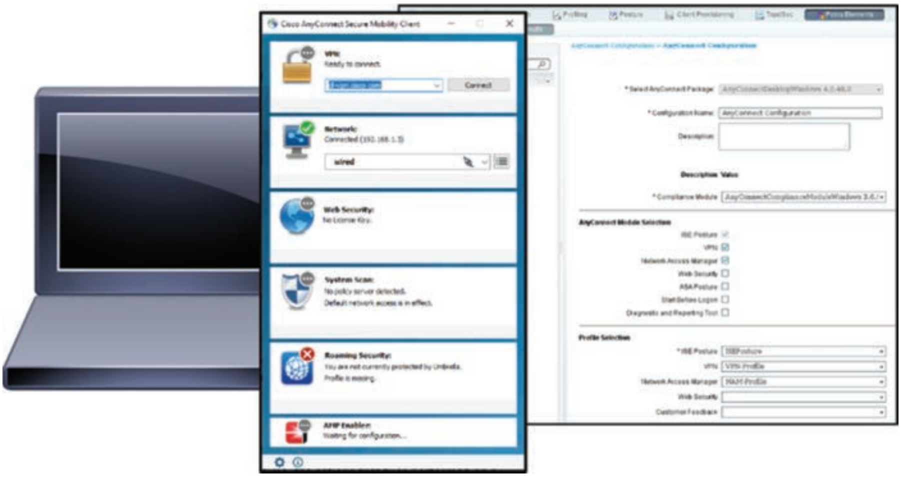
The requirements of an organization for endpoint protection software should be defined in the organization's security policy. Optimally, systems should be in place to enforce those policies as systems connect to the network.

The Cisco AnyConnect Secure Mobility Client is a multifaceted endpoint software product. It provides VPN access through Secure Sockets Layer (SSL) and also offers enhanced security through various built-in modules, such as Cisco Network Access Manager, Cisco AnyConnect ISE Agent, and Cisco AnyConnect Web Security Client. Cisco AnyConnect is available across a broad set of platforms, including Windows, macOS, Linux, iOS, and Android.

Cisco AnyConnect offers an ASA Posture module and an Identity Services Engine (ISE) Posture module, as illustrated in Figure 9-4. Both enable Cisco AnyConnect to assess compliance on endpoints for things like operating system version, antivirus software, and antispyware software installed on the endpoint. You can use these tools to restrict network access until the endpoint is compliant. The ASA Posture module is used with Cisco ASA to enforce policy for endpoints that connect to the network via remote-access VPN. The ISE posture module is used with Cisco ISE to enforce policy for internal endpoints that connect via either wired or wireless technology.

The two Posture modules perform similar functions but in different ways. The ISE Posture module performs a client-side evaluation. The AnyConnect client receives the posture requirement policy from ISE, performs the posture data collection, compares the results against the policy, and sends the assessment results back to ISE. Even though ISE determines whether the endpoint is compliant, it relies on how the endpoint evaluates the policy. In contrast, the ASA Posture module performs server-side evaluation, in which the ASA asks only for a list of endpoint attributes. AnyConnect gathers the attribute values and sends them back to the ASA. The ASA then evaluates the values to determine whether the client is allowed to create a remote-access connection to the network.

Figure 9-4 Cisco AnyConnect Posture Assessment



Cisco AnyConnect Secure Mobility Client

VPN:
Ready to connect.

Network:
Connected (192.168.1.5)
wired

Web Security:
No License Key.

System Scan:
No policy server detected.
Default network access is in effect.

Roaming Security:
You are not currently protected by Umbrella.
Profile is missing.

AMP Enabler:
Waiting for configuration...

The lists of attributes that can be examined by the two Posture modules differ, but both modules have the ability to examine an endpoint for operating system type version, antivirus software, and antispyware applications. The ASA Posture module also has the ability to examine personal firewall software. The significance of personal firewall software increases when the endpoint is outside the protective elements of the enterprise network.

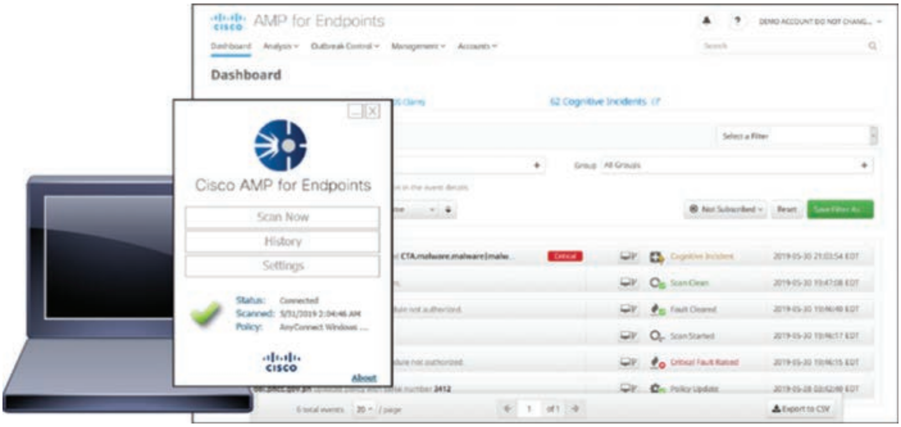
Both systems have the ability to allow connections for compliant endpoints and reject connectivity for noncompliant endpoints. They also have remediation capacities, but the two systems use different strategies. Remediation with the ASA is limited to working with the software that is already installed on the endpoint. The remediation capabilities include the ability to enable software that has been disabled, force updates for antivirus and antispyware software, and push firewall policy to the personal firewall software. Remediation with the ASA requires the advanced endpoint assessment license to be installed on the appliance. Remediation with ISE is based on quarantining the endpoint. Instead of allowing normal connectivity for the noncompliant endpoint, the endpoint is allowed very limited connectivity. The limited connectivity includes the ability to reach servers from which the required software can be obtained. After the required software has been properly installed, the Posture module is executed again. If the endpoint has reached a state of compliance, it is allowed normal access to the network.

Cisco AMP for Endpoints

Due to the nature of malware threats in current networking environments, even the best commercial products for malware detection can realistically achieve about 40% success in detection. Most enterprises implement multiple layers of protection, so malware that makes it to an endpoint defeats all the safeguards. This means that to effectively deal with malware, you must assume that at some point, the malware will make its way into your networks, and it may potentially persist for long periods of time before it is detected and acted upon.

With malware, endpoints must be protected before, during, and after attacks. Cisco AMP for Endpoints (see Figure 9-5) goes beyond point-in-time detection to provide the level of visibility and control you need to stop advanced threats that are missed by other security layers. It provides that protection across the attack continuum: before, during, and after an attack. Cisco AMP for Endpoints is an intelligent, enterprise-class advanced malware analysis and protection solution based on a telemetry model that uses big data, continuous analysis, and advanced analytics to detect, track, analyze, control, and block advanced malware outbreaks across all endpoints: PCs, Macs, mobile devices, and virtual systems.

Figure 9-5 Cisco AMP for Endpoints



Status: Connected
Scanned: 5/31/2019 2:04:46 AM
Policy: AnyConnect Windows ...

Cisco AMP for Endpoints provides cloud-based detection of malware through the Cisco Collective Security Intelligence Cloud, which is a powerful alternative to traditional malware detection and that offers these features:

- Rapid detection of known malware
- Use of cloud resources to test files with unknown dispositions
- Use of machine learning techniques to constantly keep itself up to date

AMP for Endpoints gives you a historical perspective so that you can see, over time, the actions that files have performed on a system. You can trace an infection and identify the root cause. The historical perspective gives you visibility into the following:

- **File trajectory:** Shows the hosts where files were seen
- **Device trajectory:** Shows the actions that files performed on a given host

With AMP for Endpoints you can block malicious network connections based on security intelligence feeds (IP reputation) and custom IP blacklists.

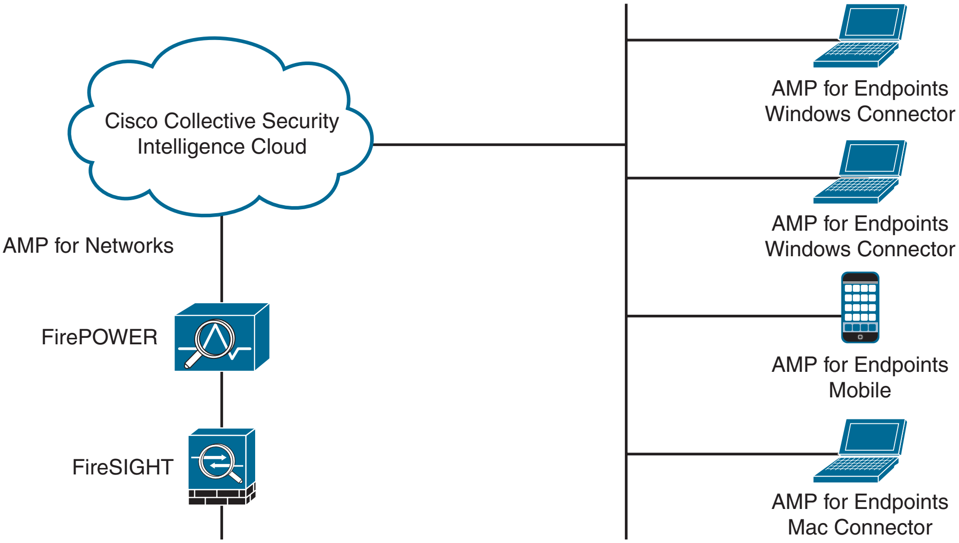
Because malware that employs stealth techniques to hide its true intent may not initially be identified as malicious, the machine learning and behavior monitoring engines in the cloud may change the disposition of a file from "unknown" to "malicious." This is known as *retrospective alerting*, or *cloud recall*. It means that Cisco AMP for Endpoints can go back to the systems where a file was previously seen, alert the client to the changed disposition, and quarantine the file.

You can deploy simple custom detections or advanced custom detections, in which you can create your own signatures for malware detection. You can create groups of hosts that can run different policies to suit the detection needs of specific environments. Cisco AMP for Endpoints also provides robust reporting tools.

As illustrated in Figure 9-6, Cisco AMP for Endpoints consists of the following elements:

- **Cisco Collective Security Intelligence Cloud:** This is where the various detection and analytics engines reside.
- **Client Connector:** This is the component that runs on the endpoints. It communicates with the cloud to send information about files and to receive file disposition information.
- **Cisco AMP for Endpoints:** You can install this application on PCs, Macs, and mobile devices to communicate with the cloud for detection of mobile malware. Cisco AMP for Endpoints is supported on various Android, Windows, Linux, and Apple operating systems.
- **AMP for Networks:** This application gives Firepower devices the ability to query the cloud to obtain file disposition information on files that are detected by the Firepower device.

Figure 9-6 Cisco AMP for Endpoints Elements


Cisco Collective Security Intelligence Cloud
AMP for Networks
FirePOWER
FireSIGHT
AMP for Endpoints Windows Connector
AMP for Endpoints Windows Connector
AMP for Endpoints Mobile
AMP for Endpoints Mac Connector

Firewall Concepts

A *firewall* is a system that enforces an access control policy between two or more security zones. There are several types of firewalls, but all firewalls should have the following properties:

- The firewall itself must be resistant to attack; otherwise, it would allow an attacker to disable the firewall or change its access rules.
- All traffic between security domains must flow through the firewall to prevent backdoor connections that could be used to bypass the firewall, violating the network access policy.
- A firewall must have traffic-filtering capabilities.

Firewalls commonly control access between security zones based on packet source and destination IP address and port.

A firewall can be a hardware appliance or a software program. Although firewalls can be placed in various locations within a network (including on endpoints), they are typically placed at the Internet edge, where they provide vital security. Firewall threat controls should be implemented at least at the most exposed and critical parts of enterprise networks. The Internet edge is the network infrastructure that provides connectivity to the Internet and acts as the gateway for the enterprise to the rest of the cyberspace. Because it is a public-facing network infrastructure, it is particularly exposed to a large array of external threats.

Firewalls are also often used to protect data centers. A data center houses most of the critical applications and data for an enterprise. The data center is primarily inward facing, and most clients are on the internal network. The intranet data center is subject to external threats but must also be guarded against threat sources inside the network perimeter.

Many firewalls also provide a suite of additional services, such as Network Address Translation (NAT) and multiple security zones. Another important service that firewalls frequently provide is VPN termination.

Firewall products have evolved to meet the needs of today's networks. From simple perimeter security with access control lists (ACLs) based on IP addresses and ports, firewalls have evolved to offer some advanced security services. The hard outer shell that firewalls provided in the past has been superseded by security capabilities that are integrated into the very fiber of the network to defend against today's multi-vector and persistent threats. To handle the current threatscape, next-generation firewalls (NGFWs) are needed.

In addition to standard first-generation firewall capabilities, NGFWs have capabilities such as the following:

- Tight integration of security functions to provide highly effective threat and advanced malware protection
- Implementation of policies based on application visibility instead of transport protocols and ports
- Provision of URL filtering and other controls over web traffic
- Provision of actionable indications of compromise to identify malware activity

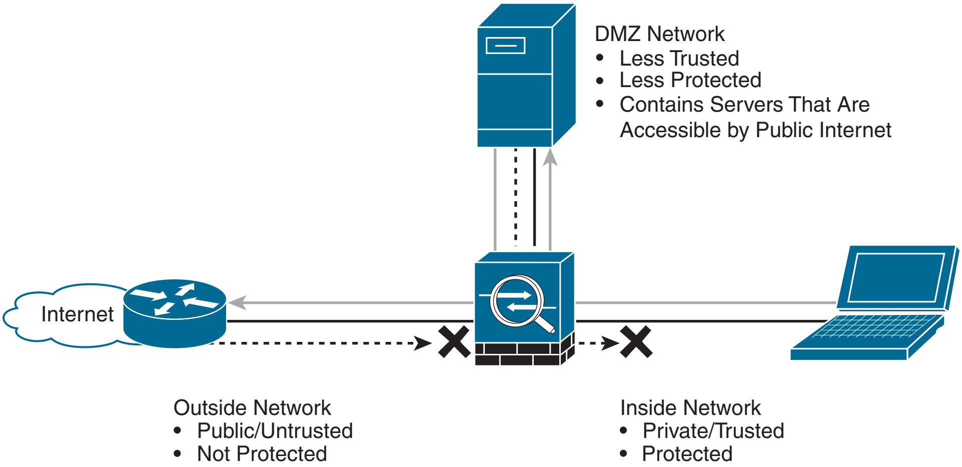
- Comprehensive network visibility
- Reduce complexity
- Smooth integration and with other security solutions

In a network, a firewall is intended to control what passes from one security zone to another. If a system is compromised in one zone, firewalls help to contain the attack to within that zone. In addition, within a network, firewalls also have the job of preventing undesirable access in the first place.

Consider the following common deployment scenario for a firewall using three zones, as illustrated in Figure 9-7:

- **Inside:** A private, trusted network
- **Outside:** The public, untrusted Internet
- **Demilitarized zone (DMZ):** A zone containing servers that are accessible by the public Internet

Figure 9-7 Firewall Zones


DMZ Network
• Less Trusted
• Less Protected
• Contains Servers That Are Accessible by Public Internet
Internet
Outside Network
• Public/Untrusted
• Not Protected
Inside Network
• Private/Trusted
• Protected

While every deployment has its own unique requirements, some policies are commonly implemented in most deployments. For example, connections from systems on the outside to systems on the inside are forbidden. Connections from the inside to the outside, on the other hand, are generally permitted. However, there may be limitations on the application protocols that are allowed from the inside to the outside. For connectivity from the outside to the DMZ, what is allowed is strictly defined. Access is only allowed to the specific ports on specific servers based on the services that are intended to be provided by the server. Also, it is common to be very restrictive in terms of the connections that can be made from servers on the DMZ to systems in other zones. If a server on the DMZ is compromised by an attacker, the compromised server should not be allowed to be used as a stepping-stone stone to the inside or to launch attacks against other systems on the outside.

Next-Generation Firewalls

Historically, the firewall has been the cornerstone of an organization's network security strategy. It was designed based on the notion that internal traffic and users are inherently trustworthy, and external traffic isn't, thus creating a trust boundary—or perimeter—between networks. This network perimeter became the logical security control point to protect the entire organization: the network, data, users, and devices. All network traffic, whether originating from the headquarters, a data center, or a remote worker, was funneled through this single control point. Since then, the ways networks are deployed and operate have changed. Many business-critical applications have moved from data centers and premises-based networks to the cloud. Branch offices are now connecting directly to the Internet, and users are accessing resources from their personal devices everywhere. NGFWs provide features such as the following to defend networks against increasingly sophisticated threats:

- **URL filtering:** URL filtering involves moving the most commonly used proxy server function into the firewall itself and may take advantage of URL classification and website reputation scores.
- **Application visibility and control:** This feature recognizes applications by analyzing data streams instead of looking at transport layer port numbers. For example, applications like Skype that are capable of hopping from one port to another can be recognized. Another example is not only recognizing Facebook but recognizing gaming within Facebook.
- **Context awareness:** Who is connecting, to what, from where, using which device, at what time? Policy can be defined, for example, to allow members of the marketing team to access Facebook for marketing purposes, but even they may not be allowed to access Facebook games. Also, what users can access via their corporate managed laptops and what they can access via their personal mobile phones may differ.
- **Intrusion prevention system:** Intrusion prevention has traditionally been handled by separate systems but can now be integrated directly into a firewall appliance, and it can be made even more powerful by utilizing the contextual awareness that is integrated into the NGFW.
- **Advanced malware protection:** This feature can provide detection, blocking, tracking, analysis, and remediation.

TrustSec

One of the limiting factors in traditional network security is that security is based on the Layer 3 addresses assigned to the endpoints. In most networks, the identity or user of the endpoint is a better indication of the trustworthiness of that endpoint. By abstracting the network addressing and using the identity associated with the endpoint for security, Cisco TrustSec makes securing the network an easier task for a network security engineer.

Cisco TrustSec simplifies the provisioning and management of secure access to network services and applications. Unlike access control mechanisms that are based on network topology, Cisco TrustSec defines policies using logical policy groupings, so secure access is consistently maintained even as resources are moved in mobile and virtualized networks. Decoupling access entitlements from IP addresses and VLANs simplifies security policy maintenance tasks, lowers operational costs, and allows common access policies to be applied to wired, wireless, and VPN access consistently.

Cisco TrustSec encompasses Security Group Tags (SGTs) and the Institute for Electrical and Electronics Engineers (IEEE) standard MAC Security (MACsec). Note that SGT tags are referred to as *Scalable Group Tags* in Cisco Software-Defined Access (SD-Access).

Cisco TrustSec classification, transport, and policy enforcement functions are embedded in Cisco switching, routing, wireless LAN, and firewall products. By classifying traffic based on the contextual identity of an endpoint instead of its source IP address, Cisco TrustSec enables more flexible access controls for dynamic networking environments and data centers. The features that are associated with SGTs on the network devices can be grouped into three categories:

- **Classification:** Classification is the assignment of an SGT to an IP address. This can be accomplished either dynamically or statically. Generally, dynamic classification is done at the access layer, and static classification is done in the data center. Dynamic classification uses the rich context data available to Cisco ISE for making policy decisions. Dynamic classification can be done using IEEE 802.1X, MAC Authentication Bypass, or Web Authentication. Static classification is generally configured on the switch to which servers are attached. Static options and configuration syntax vary by switching platform and operating system version. Options for static classification include the mapping of an IP address, a VLAN, or a port to an SGT. Also, Cisco ISE can centrally store a database of IP addresses and their corresponding SGTs. Compatible devices can download the centrally managed mappings from Cisco ISE.

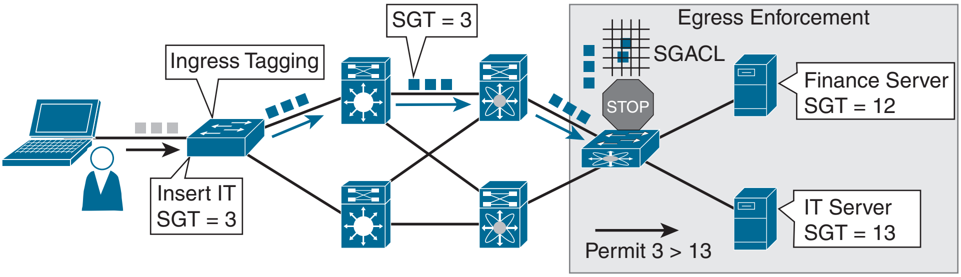
- **Transport:** Security group mappings follow the traffic through the network. This can be accomplished either through inline tagging or use of SGT Exchange Protocol (SXP). With inline tagging, the SGT is embedded in the Ethernet frame header. Not all network devices support inline tagging. SXP is used to transport SGT mappings across devices that do not support inline tagging.

- **Enforcement:** Enforcement means implementing permit or deny policy decisions based on the source and destination SGTs. This can be accomplished with security group ACLs (SGACLs) on switching platforms and security group firewalls (SGFWs) on routing and firewall platforms.

MACsec provides secure communication on wired LANs. When MACsec is used to secure the communication between endpoints on a LAN, the packets on the wire are encrypted using symmetric key cryptography, so that communication cannot be monitored or altered on the wire. When MACsec is used with Security Group Tags, it provides protection for the tag as well as the data contained in the payload of the frame.

At the point of network access, a Cisco TrustSec policy group called an SGT is assigned to an endpoint, as shown in Figure 9-8. This is done considering any of a rich set of attributes (context), including the endpoint's user, device type, posture status, and location attributes. The SGT is used to specify the endpoint access entitlements. All traffic from the endpoint carries the assigned SGT. Switches, routers, and firewalls use the SGT to make forwarding decisions. Because SGT assignments can denote business roles and functions, Cisco TrustSec controls can be defined in terms of business needs and not underlying networking detail. SGTs can span physical locations. SGTs are independent of IP address, subnet, or VLAN. SGACL-capable switches can implement access controls between systems with the same or different SGT values, even if they are on the same VLAN.

Figure 9-8 Cisco TrustSec SGT Processing



Instead of managing access control on large numbers of prefixed IP addresses, access control is managed across a relatively small number of relevant security group classifications. SGACLs define which services are accessible between two security groups. A simple matrix assigns all SGACLs to source and destination SGT pairs. One matrix provides all policy for the entire TrustSec domain. In the example shown in Figure 9-8, traffic using an SGT value of 3 is permitted access to the IT server (SGT 13) according to the SGACL configured on the data center switch.

The frames that enter the Cisco TrustSec domain are marked using an SGT. This is a unique 16-bit tag that represents the unique role of the traffic source in the network. The tag can be thought of as a privilege identifier of the source user, device, or entity. The tagging decisions can be either dynamic or static.

Dynamic tagging can be deployed with IEEE 802.1X authentication, MAC Authentication Bypass (MAB), or Web Authentication. In these access methods, Cisco ISE can push an SGT to the network access device (NAD) to be inserted into the client traffic. The SGT is applied as a permission in the authorization policy rules. This permission can be assigned in addition to or instead of an authorization profile.

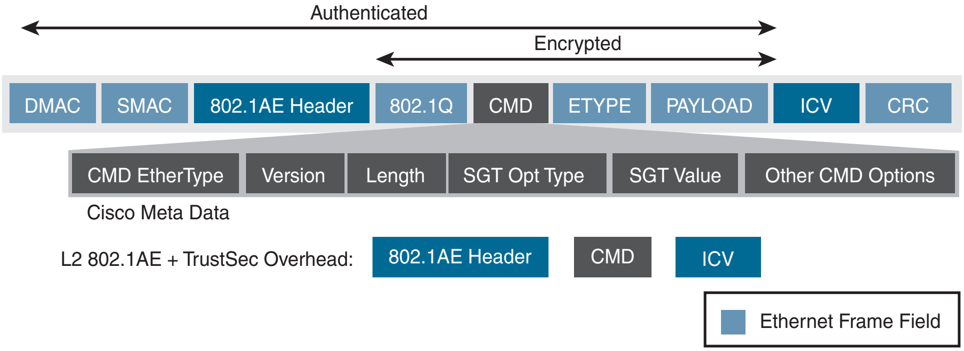
Static tagging can be configured in Cisco ISE and then downloaded to a NAD, or it can be configured directly on a NAD. Examples of static tagging include a mapping of an IP host or subnet to an SGT and the mapping of a VLAN to an SGT. Numerous other options exist, with varying support depending on the device platforms and software versions.

Because the SGT contains the security group of the source, the tag can be referred to as the *source SGT*. The destination device is also assigned to a security group (the *destination SG*), which can be referred to for simplicity as the DGT, although the actual Cisco TrustSec packet tag does not contain the security group number of the destination device. The DGT comes into play only during enforcement.

Inline SGT Transport

SGTs can be transported inline between two capable devices that are configured for Cisco TrustSec on their connecting interfaces. Using this method, the sending device imposes the SGT in the outgoing frame Ethernet header on egress, as illustrated in Figure 9-9. The receiving device reads and processes the SGT from the Ethernet header on ingress.

Figure 9-9 Cisco Meta Data (CMD) Header



The SGT is embedded in the Cisco Meta Data (CMD) header. The SGT is carried in a 16-bit field in the CMD header. Frames crossing a link between capable switches can also be protected with MACsec. A MACsec-protected frame puts the other tags and the data contained within the frame between the MACsec header and the Integrity Check Value (ICV) field. These tags and the frame data are all protected by MACsec encryption and authentication. The increase of the MTU size that results from both 802.1AE and Cisco TrustSec overhead is approximately 40 bytes.

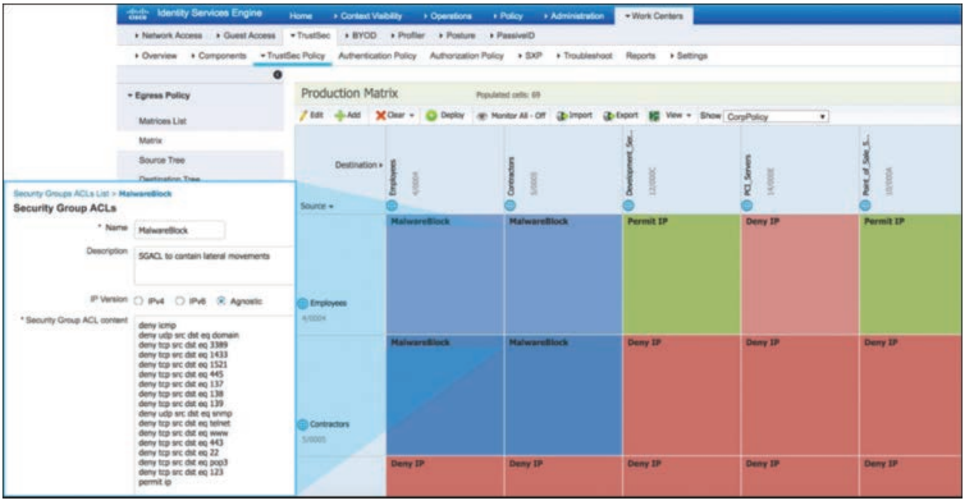
Using SGACLs, you can control access policies based on source and destination SGTs. Policy enforcement in the Cisco TrustSec domain is represented by a permissions matrix, with source security group numbers on one axis and destination security group numbers on the other axis. Each cell in the body of the matrix can contain an ordered list of SGACLs. Each SGACL specifies the permissions that should be applied to packets originating from the source security group and destined for the destination security group. It is important to note that the source and destinations are specified in the permissions matrix and not in the SGACL. Consider, for example, the SGACL entry deny TCP DST eq 21. The entry that specifies access from the source to the destination TCP port 21 is denied. There is no specification of the SGTs or DGTs in the SGACL. It is the application of the SGACL in the permissions matrix that specifies the source and destination security groups. It is also important to understand that the same SGACL can be applied to multiple source and destination security group pairs within the permissions matrix.

By applying access control between pairs of security groups, Cisco TrustSec achieves role-based, topology-independent access control within the network. Changes in network topology do not normally require a change in SGACL-based security policy. Some care must be taken to ensure the proper classification of new network resources, but the access policy based on business-relevant security groups does not change. If changes do require the creation of a new security group, then the permissions matrix increases in size by one row and one column. Policy for the new cells is defined centrally in Cisco ISE and dynamically deployed to all SGACL enforcement points.

Using role-based permissions greatly reduces the size of ACLs and simplifies their maintenance. With Cisco TrustSec, the number of ACEs configured is determined by the number of permissions specified, resulting in a much smaller number of ACEs than in a traditional IP network. Also, only a single copy of an SGACL needs to reside in the TCAM of a device, regardless of how many times the SGACL is used. The use of SGACLs in Cisco TrustSec typically results in more efficient use of TCAM resources compared with traditional ACLs.

Figure 9-10 shows the Cisco TrustSec Policy configuration page in Cisco ISE. Notice that the sources and destinations are groups, such as Employees and Contractors, and the actual SGACLs are placed in the production matrix to permit and deny certain types of traffic.

Figure 9-10 Cisco ISE TrustSec Configuration Example


Identity Services Engine
Home
Context Visibility
Operations
Policy
Administration
Work Centers
Network Access
Guest Access
TrustSec
BYOD
Profiler
Posture
PassiveID
Overview
Components
TrustSec Policy
Authentication Policy
Authorization Policy
SXP
Troubleshoot
Reports
Settings
Egress Policy
Matrices List
Matrix
Source Tree
Destination Tree
Production Matrix
Populated cells: 69
Edit
Add
Clear
Deploy
Monitor All - Off
Import
Export
View
Show
CorpPolicy
Security Groups ACLs List > MalwareBlock

Security Group ACLs

* Name: MalwareBlock

Description: SGACL to contain lateral movements

IP Version: ○ IPv4 ○ IPv6 ● Agnostic

* Security Group ACL content:

```
deny icmp
deny udp src dst eq domain
deny tcp src dst eq 3389
deny tcp src dst eq 1433
deny tcp src dst eq 1521
deny tcp src dst eq 445
deny tcp src dst eq 137
deny tcp src dst eq 138
deny tcp src dst eq 139
deny udp src dst eq snmp
deny tcp src dst eq telnet
deny tcp src dst eq www
deny tcp src dst eq 443
deny tcp src dst eq 22
deny tcp src dst eq pop3
deny tcp src dst eq 123
permit ip
```

Source ▾ / Destination ▸	Employees 4/0004	Contractors 5/0005	Development_Se... 12/000C	PCI_Servers 14/000E	Point_of_Sale_S... 10/000A
Employees 4/0004	MalwareBlock	MalwareBlock	Permit IP	Deny IP	Permit IP
Contractors 5/0005	MalwareBlock	MalwareBlock	Deny IP	Deny IP	Deny IP
	Deny IP	Deny IP	Deny IP	Deny IP	Deny IP

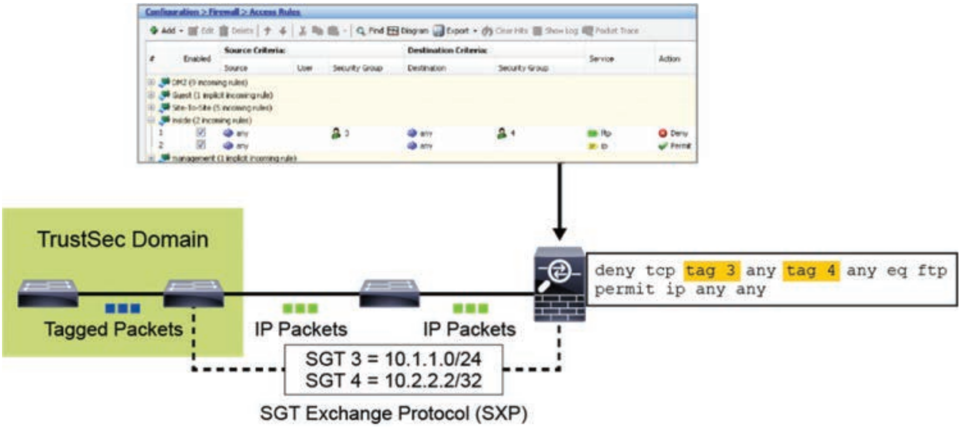
Security Group Firewall

Instead of using SGACLs, Cisco ASA and select routing platforms enforce TrustSec policy by using SGFW features. Whereas SGACL policy is centrally managed from Cisco ISE, SGFW policy is managed independently on the device configurations. SGFW capabilities were integrated into the ASA rule policy in Version 9.0 and into the Cisco IOS zone-based firewall feature in Version 15.2. Figure 9-11 illustrates the use of security group data in an access control entry on the ASA, both from Cisco ASDM and from the CLI. In this example, FTP packets sourced from SGT 3 and destined to SGT 4 are denied. All other traffic is permitted.

In the Cisco IOS zone-based policy firewall, security group statements can be used in class-map definitions. In both Cisco ASA and Cisco IOS zone-based policy firewall, security group information can be used with other traffic specification mechanisms, such as source and destination IP addresses, protocols, and port numbers. One compelling use case on the Cisco ASA is security group information used to define traffic that should be sent to a virtual IPS sensor resident in the ASA for appropriate deep packet inspection.

Cisco ASA security policy is configured with ACLs. You can create ACLs on the Cisco ASA that contain SGTs or security group names. The ASA enforces policies based on security group names or SGTs if the ASA has a security group table to map security group names to SGTs and an SGT-to-IP mapping exists.

Figure 9-11 Cisco ASA TrustSec SGFW Configuration Example



```
deny tcp tag 3 any tag 4 any eq ftp
permit ip any any
```

MACsec

MAC Security (MACsec) is an IEEE 802.1AE standards-based Layer 2 hop-by-hop encryption technology that provides data confidentiality and integrity for media access–independent protocols.

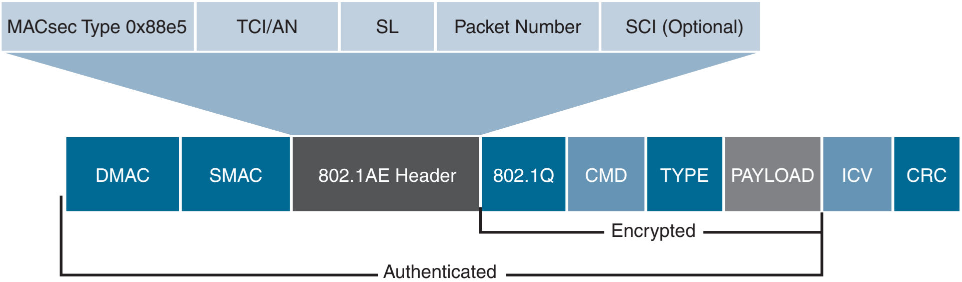
MACsec provides MAC-layer encryption over wired networks. It uses out-of-band methods for encryption keying. Required session keys are provided and encryption keys are managed using the MACsec Key Agreement (MKA) protocol. After successful authentication, MKA and MACsec are implemented using the 802.1x Extensible Authentication Protocol (EAP) framework. Only host-facing links (links between network access devices and endpoint devices such as a PC or an IP phone) can be secured using MACsec.

Depending on the policy assigned to the client connected to a secured port, a switch can accept either MACsec or non-MACsec frames. An Integrity Check Value (ICV) field encrypts and protects MACsec frames. Frames received at the switch from the client are decrypted, and the correct ICV is calculated using the session keys provided by MKA. If the calculated ICV does not match the ICV within the frame, the frame is dropped. Any frames that the switch sends to the client over the secured port are also encrypted and have an ICV field added.

Encryption keys used by the underlying MACsec protocol are managed by the MKA protocol. The MKA protocol extends 802.1x to allow peer discovery with confirmation of mutual authentication and sharing of MACsec secret keys to protect data exchanged by the peers.

The EAP framework implements MKA as a newly defined EAP over LAN (EAPoL) packet, as shown in Figure 9-12. EAP authentication produces a master session key (MSK) that is shared by both partners in the data exchange. Entering the EAP session ID generates a secure connectivity association key name (CKN). Because the switch is the authenticator, it is also the key server, generating a random 128-bit secure association key (SAK), which it sends it to the client partner. The client is never a key server and can only interact with a single MKA entity, the key server.

Figure 9-12 MACsec Frame Format


MACsec Type 0x88e5
TCI/AN
SL
Packet Number
SCI (Optional)
DMAC
SMAC
802.1AE Header
802.1Q
CMD
TYPE
PAYLOAD
ICV
CRC
Encrypted
Authenticated

The packet body in an EAPoL protocol data unit (PDU) is referred to as a MACsec Key Agreement PDU (MKPDU). MKA sessions and participants are deleted when the MKA lifetime (6 seconds) passes with no MKPDU received from a participant. For example, if a client disconnects, the participant on the switch continues to operate MKA until 6 seconds have elapsed after the last MKPDU is received from the client.

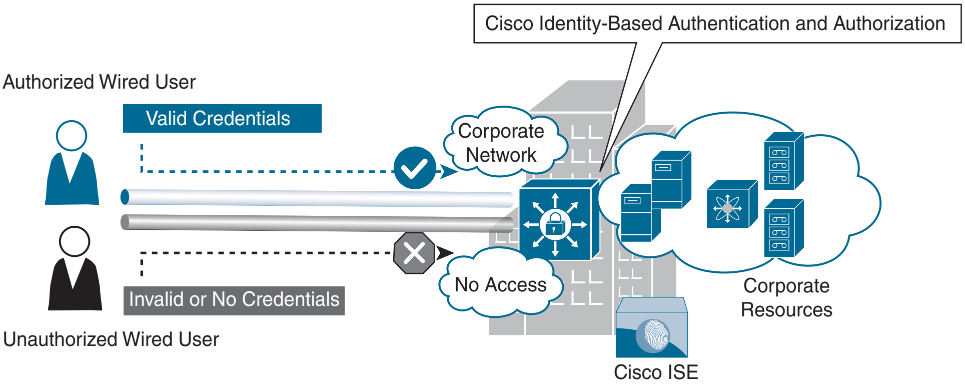
Identity Management

Instead of describing a specific product, Cisco Identity-Based Networking Services (IBNS) is a suite of services that is embedded in Cisco Catalyst switches and Cisco WLCs. Cisco IBNS supports a wide range of authentication options in which order and priority are configurable for additional flexibility. Cisco ISE uses infrastructure services provided by switches and WLCs to allow you to implement IBNS features including the following:

- 802.1X for managed devices and users
- MAC Authentication Bypass (MAB) for unmanaged or non-802.1X devices
- Web Authentication for guests or non-802.1X users

IBNS promotes authentication to access the network. Authentication serves as the basis for differentiating users and/or devices, providing varying levels of access to networked resources based on corporate access policy, as shown in Figure 9-13.

Figure 9-13 Cisco Identity-Based Networking Services (IBNS)


Cisco Identity-Based Authentication and Authorization
Authorized Wired User
Valid Credentials
Corporate Network
No Access
Invalid or No Credentials
Unauthorized Wired User
Corporate Resources
Cisco ISE

The foundation for IBNS is IEEE 802.1X, a port-based authentication and access control protocol, which can be applied at a physical switch port on a wired network or on a wireless local-area network (WLAN) on Cisco Wireless LAN Controller.

In both wired and wireless domains, clients require the installation of a Cisco 802.1X supplicant and the configuration of a native operating system supplicant or, in the case of Linux clients, the installation of an open-source supplicant.

Wireless users generally expect that they will need to authenticate before being granted access to a corporate network. As a result, populations of wireless users are good candidates for initial Cisco IBNS deployments.

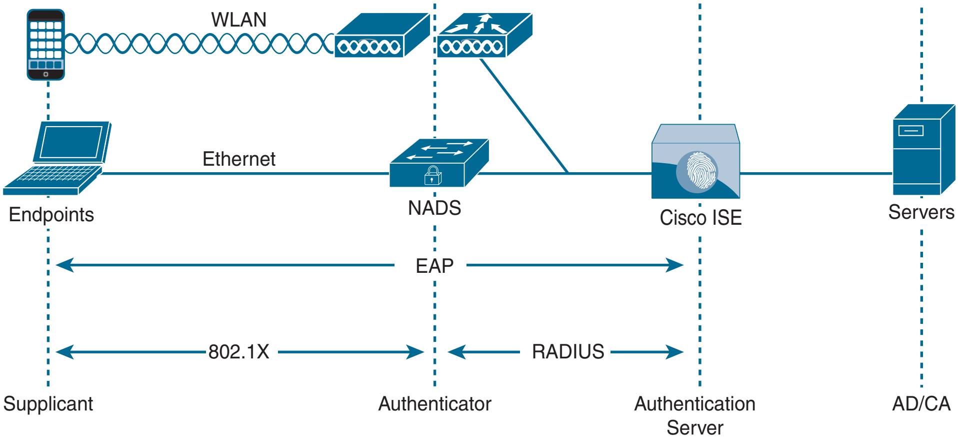
If a user or a device connecting to a network lacks the supplicant, the user or the device can authenticate using MAB or Web Authentication. Examples are guest users and simple devices, such as printers or IP cameras.

We discussed 802.1X and WebAuth in terms of wireless access on Day 10, "Wireless Client Roaming and Authentication." The next sections quickly review these concepts, focusing primarily on wired access.

802.1X for Wired and Wireless Endpoint Authentication

IEEE 802.1X provides port-based authentication. Network devices have the following roles, as shown in Figure 9-14:

Figure 9-14 802.1X Devices and Roles



- **Supplicant:** This is an endpoint 802.1X-compliant software service that communicates with network access device (NAD) authenticators to request network access.
- **Authenticator:** The authenticator controls access to the network, based on client authentication status. The objective is for endpoints to authenticate to the authentication server via EAP. NAD authenticators act as intermediaries (proxies) between client and authentication server. They communicate with endpoint supplicants via 802.1X to request identity information.

Then they communicate with the authentication server via RADIUS to verify that information. They relay authentication server responses back to the client. The authenticator acts as a RADIUS client, encapsulating and decapsulating EAP frames.

- **Authentication server:** This role performs client authentication. The authentication server validates client identity and notifies NAD authenticators of client authorization status. Because the authenticator acts as the proxy, the authentication service is transparent to the client. Cisco ISE acts as the authentication server. Credentials and certificates can be offloaded to Active Directory (via LDAP) or certificate authority servers.

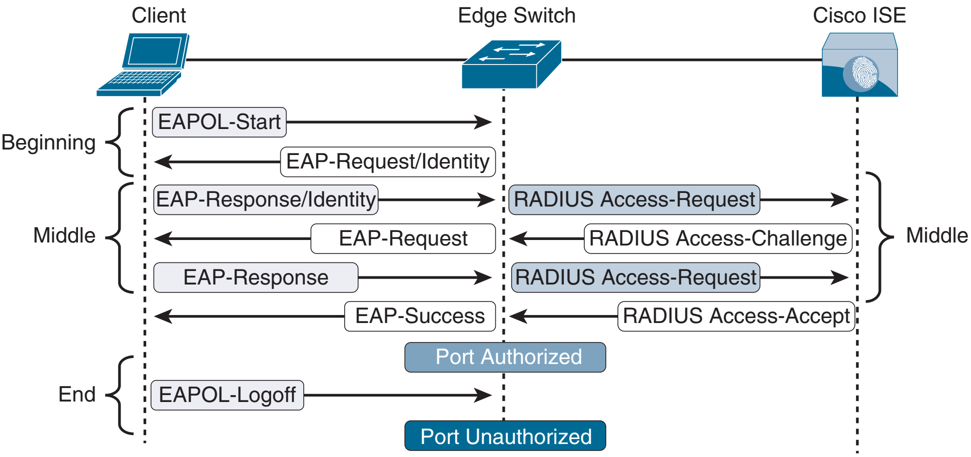
When a switch receives EAPoL packets, it frames and relays them to the authentication server. The frame's Ethernet header is stripped and then re-encapsulated into RADIUS format. The EAP fields are not modified during encapsulation, and the authentication server must support EAP in the native frame format.

When the authenticator receives EAP frames from the authentication server, the frame header is removed. The EAP frame is then encapsulated using EAPoL and sent to the supplicant.

802.1X Message Flow

Figure 9-15 illustrates the 802.1X message exchange process. Authentication can be initiated by the supplicant or the authenticator. The authenticator initiates authentication when the link state changes from down to up or periodically, as long as the port remains up and unauthenticated.

Figure 9-15 802.1X Message Flow


Client
Edge Switch
Cisco ISE
Beginning
EAPOL-Start
EAP-Request/Identity
Middle
EAP-Response/Identity
RADIUS Access-Request
EAP-Request
RADIUS Access-Challenge
EAP-Response
RADIUS Access-Request
Middle
EAP-Success
RADIUS Access-Accept
Port Authorized
End
EAPOL-Logoff
Port Unauthorized

The authenticator sends an EAP request or identity frame to request client identity. The supplicant receives the frame and sends an EAP response or identity frame. However, if during bootup the supplicant does not receive an EAP request or identity frame, it can initiate authentication. The supplicant sends an EAPoL start frame, which prompts the authenticator to request client identity.

When the supplicant provides its identity, the authenticator begins its role as the intermediary. The supplicant passes EAP frames between supplicant and authentication server until authentication succeeds or fails. If authentication succeeds, the authenticator port is authorized.

The specific exchange of EAP frames varies depending on the authentication method. Figure 9-15 depicts the three phases of the 802.1X authentication process: the beginning, the middle, and the end.

802.1X Authorization

As a result of successful authentication, Cisco ISE can perform per-user and per-group network authorization. The 802.1X framework provides authentication and authorization of clients that seek network access. The authorization features include the following:

- **VLAN assignment:** An authentication server can associate a VLAN with a particular user or group. Thus, the switch can dynamically assign a VLAN for an authenticated user. VLAN assignment is appropriate if the desired access control method is based on different VLANs (with routed access control lists [ACLs] or a firewall system that is configured egress to the VLANs). It provides strong access control and auditing. After successful 802.1X/EAP authentication, the user can be authorized to be on a specific VLAN. This dynamic VLAN is configured on the Cisco ISE RADIUS service and communicated in a RADIUS Access-Accept message. While typically used to assign a VLAN upon successful authentication, this process can also be used when authentication fails.
- **ACL assignment:** The authentication server associates an ACL with a particular user or group. It then instructs the NAD to dynamically assign the ACL to the user session. This mechanism provides very granular access control, right down to the port level. ACLs can either be named or dynamically downloaded. Named ACLs provide differentiated access for wireless users. Named ACLs are configured locally on the WLC. You merely reference the ACL in a Cisco ISE authorization policy. Of course, it is vital that the name you specify in Cisco ISE matches the name of the ACL defined in the WLC. Downloadable ACLs (dACLs) can provide different levels of access to 802.1X-authenticated users. The RADIUS server authenticates 802.1X-connected users. Based on user identity, it retrieves ACL attributes and sends them to the switch. The switch applies attributes to the 802.1X port during the user session.
- **Time-based access:** The authentication server controls each user's allowed access days and times.
- **Security group access (SGTs):** Security group access provides topology-independent, scalable access control. With security group access, the ingress switches classify data traffic for a particular role and tag the traffic with SGTs. The egress network devices evaluate the SGTs and filter packets by applying appropriate security group ACLs.

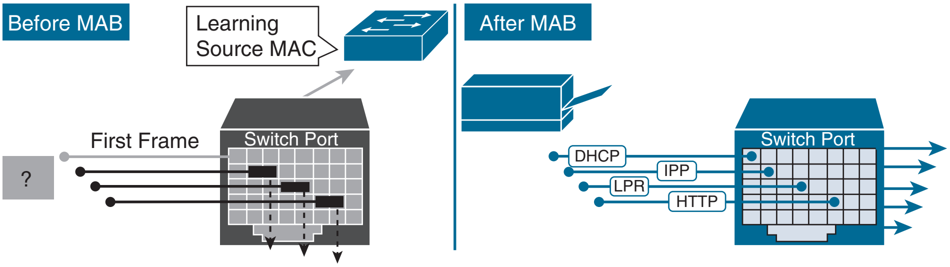
MAC Authentication Bypass

Many devices connected to the enterprise network may not support 802.1X authentication. These devices should not be exempt from authenticating before being allowed to access the network. MAC Authentication Bypass (MAB) is an alternative method that can be used when devices do not support other authentication methods.

Static MAC authentication, also known as MAB, uses a MAC address for both the username and password. This authentication is the most basic form of authentication in deployments because many devices either do not, or cannot, support 802.1X. Because MAC addresses are easily spoofed, they provide a relatively weak form of authentication, but they are a good first step for device identification.

MAB enables port-based access control by using the MAC address of the endpoint. Before MAB, the identity of the endpoint is unknown, and all traffic is blocked. The switch examines a single frame to learn and authenticate the source MAC address. After MAB succeeds and the identity of the endpoint is known, it allows all traffic from that endpoint. The switch performs source MAC address filtering to help ensure that only the MAB-authenticated endpoint is allowed to send traffic. This process is illustrated in Figure 9-16. Before MAB, all device traffic is dropped until the switch is able to authenticate the MAC address of the device. Once authenticated, traffic is authorized through the switch port.

Figure 9-16 MAC Authentication Bypass: Before and After



From the switch's perspective, the authentication session begins when the switch detects a link up on a port. The switch initiates authentication by sending an EAP Request-Identity message to the endpoint. If the switch does not receive a response, the switch retransmits the request periodically. If no response is received after the maximum number of retries, the switch lets IEEE 802.1X time out and proceed to MAB.

During the MAC address learning stage, the switch begins MAB by opening the port to accept a single packet from which it will learn the source MAC address of the endpoint. Packets sent before the port has fallen back to MAB (that is, during the IEEE 802.1X timeout phase) are discarded immediately and cannot be used to learn the MAC address.

A switch can use almost any Layer 2 and 3 packets to learn MAC addresses, with the exception of bridging frames such as Cisco Discovery Protocol, Link Layer Discovery Protocol (LLDP), Spanning Tree Protocol, and Dynamic Trunking Protocol (DTP).

After a switch learns the source MAC address, it discards the packet. Then the switch crafts a RADIUS Access-Request packet. If the MAC address is valid, the RADIUS server returns a RADIUS Access-Accept message. This message indicates to the switch that the endpoint should be allowed access to the port. Optionally, the RADIUS server may include dynamic network access policy instructions (for example, a dynamic VLAN or access control list [ACL]) in the Access-Accept message. In the absence of dynamic policy instructions, the switch simply opens the port. No further authentication methods are tried if MAB succeeds.

If the MAC address is not valid or is not allowed to access the network for policy reasons, the RADIUS server returns a RADIUS Access-Reject message. This message indicates to the switch that the endpoint should not be allowed access to the port, based on the MAC address.

Web Authentication

When guests, contractors, or employees connect to the enterprise network, they may not be able to use 802.1X authentication. An identity-based authentication that can be used in these cases is Web Authentication (WebAuth).

As discussed on Day 10, WebAuth is typically used for guest network access via Hypertext Transfer Protocol (HTTP) or HTTP Secure (HTTPS) authentication. WebAuth allows endpoints either without an 802.1X supplicant or with an 802.1X supplicant that does not authenticate to authenticate by another means.

For example, WebAuth can be used as a method of last resort when an 802.1X supplicant is not installed or when the supplicant is misconfigured or nonfunctional. If the 802.1X supplicant is nonfunctional, WebAuth could prompt the user for credentials and still provide network access. WebAuth may also be used for guest users who have an 802.1X supplicant but lack a user account in the appropriate identity store.

WebAuth can be implemented in both wired and wireless environments. With WebAuth, users connect to the network, and their initial attempt to access a URL is automatically redirected to an authentication web page.

When wired or wireless guest users first connect, they have limited access. This can be defined in Cisco ISE through a dACL or a VLAN on a switch or the named ACL on a wireless LAN controller (WLC). At this initial stage, the authorization assignment permits only WebAuth traffic. All other traffic is redirected to the WebAuth service.

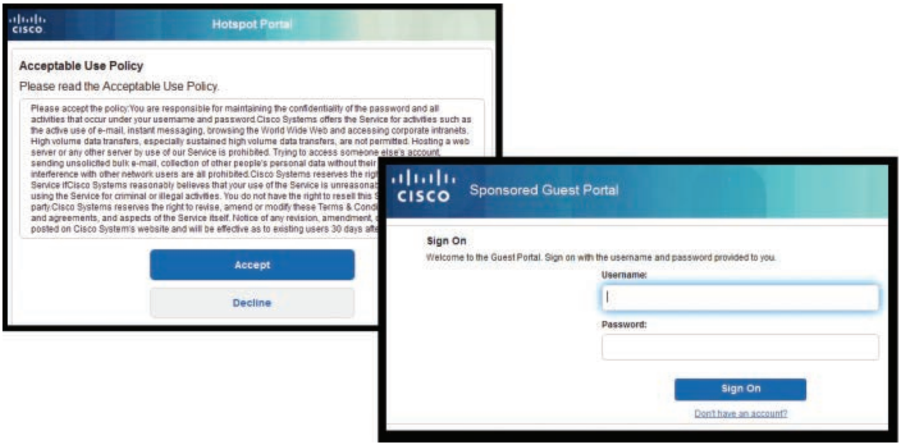
To gain full access, guest users browse via HTTP or HTTPS to a DNS-resolvable website, such as http://guests.cisco.com. The NAD intercepts the HTTP/HTTPS request and redirects it to the guest user login portal. After successful authentication, the user is assigned an appropriate authorization profile. This profile allows appropriate access to enterprise resources.

WebAuth using Cisco ISE supports five scenarios:

- **NAD with Central WebAuth:** This scenario applies to wired and wireless network access devices. In this scenario, the user is redirected to the Cisco ISE web service for authentication. The authentication is performed on Cisco ISE. Cisco ISE sends a change of authorization (CoA) to the NAD after authentication.
- **WLC with Local WebAuth:** In this scenario, the wireless user logs in and is directed to the WLC. The WLC then redirects the user to the guest portal. The guest portal prompts the user for a username and a password and requires the acceptance of an acceptable use policy (AUP), as shown in Figure 9-17. When this process is complete, the user's browser is redirected back to the WLC to log in again. The WLC authenticates the user via RADIUS and then redirects the client browser to the original destination.
- **Wired NAD with Local WebAuth:** In this scenario, the guest user login portal redirects the wired guest user login to the switch. The login request is in the form of an HTTPS URL that is posted to the switch and contains the user credentials. The switch receives the user login request and authenticates the user through a RADIUS server that points to Cisco ISE.

- **Device registration WebAuth:** In this scenario, the user connects to the network with a wireless connection. An initial MAB request is sent to the Cisco ISE node. If the user MAC address is not in the endpoint identity store, Cisco ISE responds with a URL redirection authorization profile. The URL redirection presents the user with an AUP acceptance page when the user attempts to browse to any URL.
- **Cisco EasyConnect:** The Cisco ISE EasyConnect feature enables enterprises to implement identity-based network access without the need for 802.1X. No supplicants or supplicant configurations are needed on endpoints. An EasyConnect session, which is similar to the WebAuth flow, starts with MAP. ISE learns about an endpoint's location, MAC address, and IP address via an initial MAB session. This initial MAB session is authorized with limited access from ISE to enable a Windows Active Directory–managed endpoint to perform a Windows domain login. Upon successful domain login, the user ID-to-IP address mapping from the Active Directory (AD) domain controller is sent to ISE and merged with the initial MAB session. After the user ID and its AD group membership are resolved, ISE changes the authorization to permit additional access.

Figure 9-17 WebAuth Login and AUP Screens



Acceptable Use Policy

Please read the Acceptable Use Policy.

Please accept the policy:You are responsible for maintaining the confidentiality of the password and all
activities that occur under your username and password.Cisco Systems offers the Service for activities such as
the active use of e-mail, instant messaging, browsing the World Wide Web and accessing corporate intranets.
High volume data transfers, especially sustained high volume data transfers, are not permitted. Hosting a web
server or any other server by use of our Service is prohibited. Trying to access someone else's account,
sending unsolicited bulk e-mail, collection of other people's personal data without their
interference with other network users are all prohibited.Cisco Systems reserves the rig
Service ifCisco Systems reasonably believes that your use of the Service is unreasonab
using the Service for criminal or illegal activities. You do not have the right to resell this S
party.Cisco Systems reserves the right to revise, amend or modify these Terms & Cond
and agreements, and aspects of the Service itself. Notice of any revision, amendment, o
posted on Cisco System's website and will be effective as to existing users 30 days afte

Sign On

Welcome to the Guest Portal. Sign on with the username and password provided to you.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	25

Day 8

Infrastructure Security

ENCOR 350-401 Exam Topics

Security

- Configure and verify device access control
  - Lines and password protection
  - Authentication and authorization using AAA
- Configure and verify infrastructure security features
  - ACLs
  - CoPP

Key Topics

Today we review different tools and technologies that can be deployed in a network to provide infrastructure security. First, we look at password-protecting devices and configuring authentication, authorization, and accounting (AAA). Then we focus on deploying access control lists (ACLs) and enabling control plane policing (CoPP).

Secure Access Control

The AAA framework enables configuration of AAA functions in a uniform yet flexible way. It is important to identify users before they access the network; this is the authentication phase. The next step is to determine what users are capable of doing and what restrictions apply to them; this is the authorization phase. Finally, you want to keep the information about user activities for future auditing, billing, or reporting purposes; this is the accounting phase.

Today we review each of the AAA components, compare possible authentication options that you can implement on Cisco IOS Software, and look at how centralized implementation of AAA services by using a TACACS+ or RADIUS server contributes to better scalability, manageability, and control in an enterprise environment. Today we focus on the configuration of authentication by using a centralized server and a local database. We also consider the limitations of TACACS+ and RADIUS and examine implementation examples. Before delving into AAA, we briefly review how to secure device access.

Securing Device Access

It is essential that a network device's console, auxiliary, and vty lines be secured using strong passwords. Access to a device's command-line interface (CLI) can be provided by physical connections to the console port or AUX port. This type of direct serial connection to a device can facilitate out-of-band (OOB) management access. The CLI can also be reached through networking protocols via a device's vty lines. In the past, Telnet was commonly used as a protocol for accessing a network device's CLI. While the use of Telnet does persist, its popularity is waning due to security concerns. Telnet does not provide any privacy. Data is sent in plaintext, and authentication credentials are also sent in plaintext. An attacker who can sniff a Telnet session can capture the administrator's username and password. Therefore, Secure Shell (SSH) has become the standard for CLI network access. SSH enables a client to verify the identity of a server (the network device), which helps prevent man-in-the-middle attacks and protects the authentication credentials and the data through the use of encryption.

It is possible to access a device's GUI via HTTP and HTTPS. Just as SSH is preferred over Telnet, HTTPS is preferred over HTTP from a security perspective. HTTPS allows a client to verify the identity of a network device and protects the administrator's credentials and all data passing over the network.

Ideally, all passwords associated with routers should be stored on an external AAA server; however, it is often necessary to locally store passwords on a router. If someone could see a router's running configuration, that person would be able to see any passwords that were in plaintext. Therefore, a best practice security recommendation is to encrypt any passwords that appear in a router's configuration.

Several passwords in Cisco IOS should be encrypted (or represented as hash values), including the enable secret password, line password, and username password.

Enable Secret Password

The enable secret password can be used to give a network engineer full privileges on a router. This password is configured with the **enable secret** *password* global configuration mode command. By default, the password then appears in a router's running configuration as a Message Digest 5 (MD5) hash value.

Starting with IOS Version 15.3(3), new stronger algorithm types are available for the enable secret password. To change the algorithm type used to hash a password configured using the **enable secret** command, use the **enable algorithm-type** command in global configuration mode.

When viewed in the running configuration, a password hashed with SHA-256 appears with the number **8** preceding the hash value, and a password hashed with SCRYPT appears with the number **9** preceding the hash value. The SCRYPT algorithm is considered more resistant to brute-force attacks than SHA-256. The MD5 algorithm is now considered insecure and should also be avoided. If MD5 is used, the number **5** appears before the hash value.

Example 8-1 shows the enable secret algorithm options that are available on Cisco devices and provides an example of the enable secret password **Cisco123** being configured with the SCRYPT hashing algorithm.

Example 8-1 Configuring Enable Secret Algorithm Types

```
Router(config)# enable algorithm-type ?
  md5     Encode the password using the MD5 algorithm
  scrypt  Encode the password using the SCRYPT hashing algorithm
  sha256  Encode the password using the PBKDF2 hashing algorithm

Router(config)# enable algorithm-type scrypt secret Cisco123
Router(config)# do show running-config | section enable secret
enable secret 9 $9$9R6fhUxoNVKn5X$./Iej24CapNmPzTt74iq8Yx7Ic6dr.2os08FaP88svs
```

Line Password

A line password is used to authenticate a user attempting to log in to one of a router's lines, such as a vty line, the console line, or the auxiliary line. You can define a line password in line configuration mode with the **password** *password* command; however, at that point, the password still shows up in the router's running configuration in plaintext. To encrypt that password, you can issue the **service password-encryption** command in global configuration mode. Unfortunately, this type of encryption is not very strong. It uses the Vigenère cipher and is also known as Type 7 encryption. While this type of encryption can protect passwords from a casual observer who happens to catch a glimpse of the password, it can easily be deciphered (using freely available utilities on the Internet) if someone were to come into possession of a router's running configuration. Therefore, Cisco recommends configuring username/password combinations (as discussed in the next section) and requiring those credentials to be entered before accessing one of the router's lines.

In Example 8-2, the line password **cisco** is configured for the console line. The **login** command makes it possible for someone to log in to the console port by supplying the configured password as the only authentication credential. However, in the running configuration in Example 8-2, the password was not encrypted, and the **service password-encryption** command is entered to encrypt the password. Unfortunately, the type of encryption used in this example is Type 7, which is very weak. A better method would be to authenticate users based on a username/password combination, where the password appears in the running configuration as an encrypted hash value. The **exec-timeout 5 0** command prevents users from remaining connected to a console port when they leave a station. In Example 8-2, when no user input is detected on the console for 5 minutes, the user that is connected to the console port is automatically disconnected.

Example 8-2 Configuring Line Passwords

```
Router(config)# line con 0
Router(config-line)# password cisco
Router(config-line)# login
Router(config-line)# exec-timeout 5 0
Router(config-line)# do show running-config | section line
line con 0
 password cisco
 login
<...output omitted...>
```

```
Router(config)# service password-encryption
Router(config)# do show running-config | section line
line con 0
 password 7 1511021F0725
<...output omitted...>
```

Username and Password

Instead of just requiring a password to log in to a router, you can create a username/password combination that a network engineer must enter to gain access to the router. Usernames can also be configured with various privilege levels (where a privilege level of 15 indicates a full set of privileges). These privilege levels can be used to control what Cisco IOS commands a user can execute.

You can populate a locally stored user database with the **username** *username* **privilege** *privilege-level* **secret** *password* command. By default, this causes an MD5 hash of the password to appear in the router's running configuration, which is more secure than Type 7 encryption but could still be improved upon by either using Type 8 or Type 9 encryption.

Example 8-3 shows the configuration of four usernames. The first is an example of a username/password combination using a plaintext password. This should be avoided in production networks. Next, the username of **admin** with the MD5 secret password **Cisco123** is configured with a privilege level of 15. Level 15 is the highest level of privilege on a Cisco IOS device and has access to all commands.

Example 8-3 Configuring Usernames and Passwords

```
Router(config)# username badexample password badpassword
Router(config)# username admin privilege 15 secret Cisco123
Router(config)# username tier1 algorithm-type scrypt secret tier1secret
Router(config)# username tier2 algorithm-type sha256 secret tier2secret

Router(config)# line vty 0 4
Router(config-line)# login local
Router(config-line)# do show running-config | include username
username admin privilege 15 secret 5 $1$Cmuf$AUhxqg9U8vtmyXxnIvws90
username tier1 secret 9 $9$SDMCRTSdeALwnn$wCVReuYIqhfiJqdQ.o4SoxZL6pd
  TdodYgZBcVmqKyws
username tier2 secret 8 $8$fKF1lRle.c6i9X$NTeH9114hDNQpTB4ZN7yoJtcQcGYL0VSYn47bMK
  6nOE
username badexample password 7 011107004B0A151C36435C0D
```

As with the **enable secret** command, it is possible to apply Type 8 or Type 9 encryption to the secret password associated with a username. In Example 8-3, the user **tier1** has been configured with the Type 9 password **tier1secret**, and the **tier2** user has been configured with the Type 8 password **tier2secret**.

The **login local** command issued in line configuration mode tells the vty lines to use the router's local user account database for authentication. The previously configured line password **cisco** is no longer used.

When viewed in the running configuration, notice that all passwords are hashed according to their individual encryption algorithms: 5 for MD5, 9 for Type 9 (SCRYPT), 8 for Type 8 (SHA256), and 7 for Vigenère.

The local username/password database can also be used when authenticating an HTTP or HTTPS connection to the device. In that case, the **ip http authentication local** command must be entered in global configuration mode.

AAA Framework Overview

AAA is a framework through which network access control policy is enforced on a networking device. The three main functions of the AAA model are configurable in a consistent manner:

- **Authentication:** Authentication is the process of identifying a user before the user is allowed access to a protected resource. A user presents valid credentials, which are then compared with the information in the user database. In addition, the authentication component may offer other services, depending on the security protocol selected (for example, additional challenge and response, messaging support, encryption). Authentication defines and confirms the "Who you are?" aspect.
- **Authorization:** After the user gains access to the network, authorization is performed. Authorization allows you to control the level of access that users have. For example, via authorization, you can define which privilege EXEC commands are available to the user, and you can control remote access by allowing a user to use protocols such as Point-to-Point Protocol (PPP). User capabilities are defined by a set of attribute/value pairs that are associated with the user or user group. These pairs may be stored locally on the device or on a centralized Terminal Access Controller Access-Control System Plus (TACACS+) or Remote Authentication Dial-In User Service (RADIUS) server. When a user tries to perform some specific function (for example, configure an IP address on an interface), the AAA engine on the device queries the authorization server for that specific attribute and user. Based on the reply from the server (that is, the value of the user attribute in question), the user is allowed or not allowed to perform that specific function. Authorization defines the "What you can do as an authenticated user" aspect.
- **Accounting:** Accounting, which is performed after authentication, enables you to collect information about the activity and resource consumption for a user. It allows you to keep track of user logins, commands executed by the user, session durations, bytes transferred, and so on. A network device sends this information to the accounting server in the form of attribute/value pairs. User activity information from all devices in the network is thus located in one central place. This information can then be used for billing, auditing, and reporting purposes. Accounting defines the "What you did and when" aspect.

Authentication can be valid without authorization and accounting. Authorization and accounting, however, cannot be performed without authentication.

Implementing the AAA model across the enterprise brings the following advantages:

- **Increased flexibility and granular control of access configuration:** Although Cisco IOS Software offers possibilities to configure local usernames and passwords, line passwords, and enable secret passwords without the use of AAA, these methods do not offer satisfactory flexibility and control. AAA offers authorization flexibility on a per-command or per-interface level, which is unavailable with local credentials.

- **Scalability:** Storing usernames and passwords in a local database on a device may be an appropriate solution for a small network with a few users. As the network grows, managing many users on multiple devices becomes highly impractical and error prone, with a lot of administrative burden. When credentials become outdated or missing, several network administrators may use a single username, and it becomes impossible to track activities back to a particular user. The AAA model is the only solution that scales well.

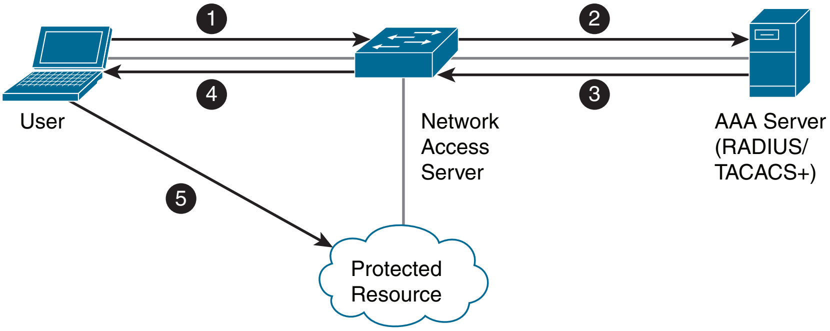
- **Standardized and proprietary authentication methods:** The AAA framework supports the RADIUS protocol, which is an industry open standard. This protocol ensures interoperability and allows flexibility because you can mix and match different vendors. For example, you can have a network built with Cisco equipment and have a AAA server from another vendor, or you can even build your own custom AAA server that is based on RADIUS.

- **Redundancy:** You can specify multiple servers when configuring an authentication option on the method list and combine them into a server group. In the event of a server failure, the AAA engine on the device can continue to query the next server from the server group.

RADIUS and TACACS+

RADIUS and TACACS+ are AAA protocols. Both of them use a client/server model, as illustrated in Figure 8-1. A user or machine sends a request (step 1 in the figure) to a client, which is also called a *network access server* (*NAS*) or *remote access server* (*RAS*). Typically, a NAS is a router, a switch, a firewall, or an access point. The client then communicates (steps 2 and 3) with a server by exchanging RADIUS or TACACS+ messages. If authentication is successful, the user is granted (step 4) access to a protected resource (step 5), such as a device CLI or network. Cisco implements the AAA server functionality in the Cisco Secure Access Control Server (ACS) and the Cisco Identity Services Engine (ISE).

Figure 8-1 Client/Server Model



RADIUS is an open standard protocol developed by Livingston Enterprises. It is described in RFCs 2865 (authentication and authorization) and 2866 (accounting). RADIUS uses UDP port 1812 for authentication and authorization, thus combining authentication and authorization into one service. For the accounting service, RADIUS uses UDP port 1813. Communication between the NAS and RADIUS server is not completely secure: Only the password portion of the RADIUS packet header is encrypted.

TACACS+ is a Cisco-proprietary protocol that is not compatible with older versions such as TACACS or XTACACS, which have been deprecated. TACACS+ allows for greater granularity through total separation of the three AAA functions. TACACS+ uses TCP port 49; thus, reliability is ensured by the transport protocol itself. The entire TACACS+ packet is encrypted, so communication between the NAS and the TACACS+ server is completely secure.

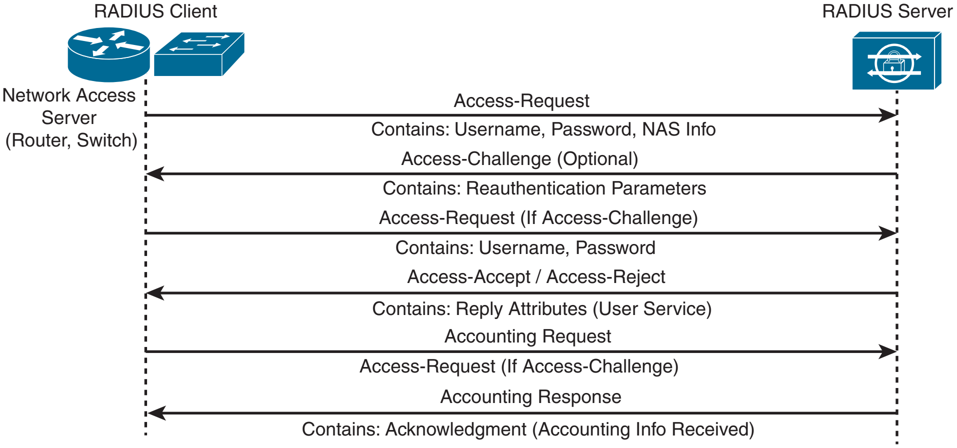
There are two main uses for AAA in networking:

- **Device administration:** Controlling access to who can log in to a network device console, Telnet session, or SSH session is AAA for device administration. While AAA for device administration can seem similar to network access AAA, it has a completely different purpose and requires different policy constructs. TACACS+ falls into this category of AAA.
- **Network access:** Securing network access can provide the identity of the device or user before permitting the entity to communicate with the network. This is AAA for secure network access. RADIUS usually falls into this category of AAA.

RADIUS Packet Flow

The authentication process between the NAS and the RADIUS server starts when a client sends a login request in the form of an Access-Request packet, as shown in Figure 8-2. This packet contains the username, encrypted password, NAS IP address, and NAS port number.

Figure 8-2 RADIUS Packet Flow


RADIUS Client
RADIUS Server
Network Access Server (Router, Switch)
Access-Request
Contains: Username, Password, NAS Info
Access-Challenge (Optional)
Contains: Reauthentication Parameters
Access-Request (If Access-Challenge)
Contains: Username, Password
Access-Accept / Access-Reject
Contains: Reply Attributes (User Service)
Accounting Request
Access-Request (If Access-Challenge)
Accounting Response
Contains: Acknowledgment (Accounting Info Received)

When the server receives the query, it first compares the shared secret key that is sent in the request packet with the value that is configured on the server. When the shared secrets are not identical, the server silently drops the packet. This action ensures that only authorized clients are able to communicate with the server. If the shared secrets are identical, the packet is further processed by comparing the username and password inside the packet with the information in the database. If a match is found, the server returns an Access-Accept packet with a list of attributes (for example, IP address, ACL for NAS) to be used with this session, in the form of attribute/value pairs. If, on the other hand, a match is not found, the RADIUS server returns an Access-Reject packet. It is important to keep in mind that the authentication and authorization phases are combined in a single Access-Request packet.

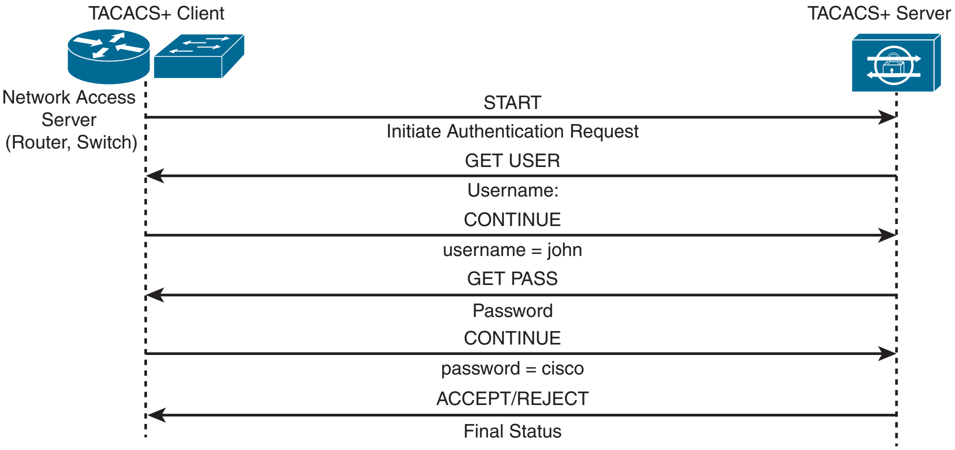
During the authentication and authorization phases, the RADIUS server may request an optional Access-Challenge message in order to collect additional data (for example, PIN, token card), thus further verifying the identity of the client.

The accounting phase occurs separately after the authentication and authorization phases, and it involves the Accounting-Request and Accounting-Response messages.

TACACS+ Packet Flow

Communication between the NAS and the TACACS+ server starts with an established TCP connection. Figure 8-3 shows the TACACS+ packet exchange process that occurs after a TCP connection has been established.

Figure 8-3 TACACS+ Packet Flow



The NAS contacts the TACACS+ server to obtain a username prompt, which is then displayed to the user. The user enters a username, which is forwarded to the server. The server prompts the user again, this time for a password. The password is sent to the server, where it is validated against the database (local or remote). If a match is found, the server sends an Accept message to the client, and the authorization phase can begin (if configured on the NAS). If, on the other hand, a match is not found, the server responds with a Reject message, and any further access is denied.

Authentication Options

Before a user receives access to a network or device resource, you need to verify the identity of the user. This process is called *authentication*.

Authentication can be based on something that a user knows (such as a username and password), something that a user has (such as a digital certificate issued by a certificate authority), or something that a user is (such as a biometric feature, such as fingerprint or eye retina).

Whichever method is used, the information provided by the user is compared with the information stored in the authentication database. If a match exists, the user is granted access to the network. The authentication database may be stored either locally on a network device or on a centralized server.

The AAA model in Cisco IOS Software allows you to configure both local and remote authentication. Configuration is done with the **aaa authentication** command, as shown here:

```
Router(config)# aaa authentication service {default | list-name} method1 [method2 . . . ]
```

You need to specify a service type for which to apply the authentication configuration. The service type might be **login** for CLI login authentication (console, Telnet, SSH), **ppp** for PPP connections, **dot1x** for 802.1X-compliant authentication with RADIUS, and so on.

Next, you need to specify a method list. You can specify a default method list with the **default** keyword. The default method list applies to any interface, line, or service if a more specific named method list is not defined. It is typically used in smaller or medium-size environments, where there is a single shared AAA infrastructure. Also, named method lists can be specified. When it is defined, the named method list must be explicitly applied to an interface, line, or service, and it overrides the default method list. You can define named method lists according to your needs.

Finally, you need to specify a single method or a list of authentication methods. Multiple authentication methods are available, such as **local** for local database authentication and **group** to configure groups of remote RADIUS or TACACS+ servers. When multiple authentication methods are configured, additional methods of authentication are used only if the previous method returns an error but not if it fails. To specify that the local username database should be used for authentication, use the **local** or **local-case** method. To specify that the enable password should be used for authentication, use the **enable** method. To specify that the authentication should succeed even if all methods return errors, use **none** as the final method in the command line. Using the **none** authentication method is discouraged in production environments because it allows access without successful authentication.

Enabling AAA and Configuring a Local User

To enable AAA, you need to configure the **aaa new-model** command in global configuration mode. Until this command is enabled, all other AAA commands are hidden. The **aaa new-model** command immediately applies local authentication to all lines and interfaces (except the console line). To avoid being locked out of the router, you should define a local username and password before starting the AAA configuration.

You can use different authentication options. You can use external authentication servers such as RADIUS or TACACS+, or you can specify a local database with the **username** command, as shown in Example 8-4. In any case, it is a good practice to configure a local username to serve as a backup in the event that the external servers fail. The keyword **password** and the keyword **secret** are available. The secret will be stored in the configuration in a more secure fashion than would a plaintext password.

Example 8-4 Configuring the Local Database

```
Router(config)# username admin secret adminpass
Router(config)# aaa new-model
Router(config)# aaa authentication login default local
```

To use the local database for AAA authentication, you must specify the **local** keyword in the **aaa authentication** command.

To use the local database as a backup AAA authentication option only, you need to tell the device to use the local database only if all other authentication methods are unavailable. To do so, you must specify the **local** keyword at the very end of the authentication methods list with the **aaa authentication** command.

Configuring RADIUS for Console and vty Access

RADIUS AAA configuration starts with the configuration of a RADIUS server. Example 8-5 shows a RADIUS server being configured using the name RADSRV1.

Example 8-5 Configuring RADIUS for Console and vty Access

```
Router(config)# radius server RADSRV1
Router(config-radius-server)# address ipv4 10.255.255.101 auth-port 1812
  acct-port 1813
Router(config-radius-server)# key SecretRAD
Router(config-radius-server)# exit
Router(config)# aaa group server radius RADSRVGROUP1
Router(config-sg-radius)# server name RADSRV1
Router(config-sg-radius)# exit
Router(config)# aaa authentication login SRVAUTH group RADSRVGROUP1 local
Router(config)# line con 0
Router(config-line)# login authentication SRVAUTH
Router(config-line)# line vty 0 4
Router(config-line)# login authentication SRVAUTH
```

You need to specify the hostname or the IP address of the server. Optionally, you can specify a custom port number for the UDP communication if your RADIUS server is listening on nondefault ports. Port numbers for authentication and accounting differ. The **SecretRAD** key string specifies the authentication and encryption key used between the access device and RADIUS server. This value must match on the two devices.

Next, you add the RADIUS server to a server group. You can add multiple RADIUS servers to a group if they were previously defined with the **radius server** command. In Example 8-5, a group called RADSRVGROUP1 is defined, and the RADSRV1 server is added to it.

Example 8-5 also shows the configuration of login authentication using a named method list called SRVAUTH. The server group RADSRVGROUP1 is set as the primary authentication option, and the local user database is set as a backup.

Finally, the example shows the SRVAUTH method list being applied to the console and the vty 0 4 lines. Recall that named method lists, unlike the default list, must be applied to specific lines in order to take effect.

Configuring TACACS+ for Console and vty Access

The TACACS+ AAA configuration is nearly identical to the RADIUS configuration. First, you need to configure the TACACS+ server, as shown in Example 8-6.

Example 8-6 Configuring TACACS+ for Console and vty Access

```
Router(config)# tacacs server TACSRV1
Router(config-tacacs-server)# address ipv4 10.255.255.102
Router(config-tacacs-server)# key SecretTAC
Router(config-tacacs-server)# exit
Router(config)# aaa group server tacacs+ TACSRVGROUP1
Router(config-sg-tacacs)# server name TACSRV1
Router(config-sg-tacacs)# exit
Router(config)# aaa authentication login default group TACSRVGROUP1 local
```

You need to specify the hostname or the IP address of the server. Optionally, you can specify a custom port number for the TCP communication if your TACACS+ server is listening on a nondefault port. The key string specifies the encryption key to be used for encrypting all traffic between the access device and the TACACS+ server. This value must match on the two devices.

Next, the TACACS+ server is added to a server group. You can add multiple TACACS+ servers to a group if they were previously defined with the **tacacs server** command. You would do this to ensure redundancy to cope with a failure of the primary TACACS+ server.

Example 8-6 shows the configuration of a TACACS+ server with IP address 10.255.255.102 and shared secret key SecretTAC as a part of the server group named TACSRVGROUP1.

Like the previous RADIUS example, this TACACS+ example shows the configuration of a default login authentication list; in this case, the server group TACSRVGROUP1 is a primary authentication option, and the local user database is a backup.

The default method list is automatically applied to all lines except the lines that have a named method list that is explicitly defined.

Configuring Authorization and Accounting

Once AAA is enabled on a Cisco IOS device and the **aaa authentication** command is configured, you can optionally configure the dependent AAA functions **aaa authorization** and **aaa accounting**. After a user is authenticated, the device allows access to certain services or commands, based on the user's privilege level. Authenticated users are put at the EXEC level by default. Certain commands, such as **show interface**, are available at the EXEC level. Other commands, such as **configure terminal**, are accessible only if the user can move into the privileged EXEC or enable mode.

Authorization provides a means of granting specific users the ability to perform certain tasks. Like authentication, authorization is performed by querying external RADIUS or TACACS+ servers. If the authorization server has an entry for a user and a service or command, the device allows the user to perform that task.

You configure authorization by first defining any RADIUS or TACACS+ servers that will be used. These normally are defined as part of the authentication configuration and do not need to be redefined for authorization.

Next, define a method list of authorization methods that will be tried in sequence, using the following global configuration command:

```
Router(config)# aaa authorization {commands | config-commands | configuration | exec
  | network | reverse-access | console} {default | list-name} method1 [method2 ...]
```

Here you can specify the function or service that needs authorization with one of the following keywords:

- **commands:** The server must return permission to use any router command at any privilege level.
- **config-commands:** The server must return permission to use any router configuration command.
- **exec:** The server must return permission for the user to run a router EXEC session. The server also can return the privilege level for the user so that the user can be put into privileged EXEC (enable) mode immediately, without having to type in the enable command.

You can identify the method with a descriptive name (*list-name*) if you are configuring more than one list. Otherwise, a single unnamed list is called **default**. The authorization methods are then listed in the order in which they will be tried.

Example 8-7 shows the configuration of a named method list called MYTACAUTH for EXEC and level 15 command authorization, as well as authorization for access to configuration mode. Notice the **if-authenticated** keyword at the end of the first line. This says that a user who is authenticated will be put into EXEC (enable) mode immediately.

Example 8-7 Configuring AAA Authorization

```
Router(config)# aaa authorization exec MYTACAUTH group TACSRVGROUP1 local
  if-authenticated
Router(config)# aaa authorization commands 15 MYTACAUTH group TACSRVGROUP1 local
Router(config)# aaa authorization config-commands
Router(config)# aaa authorization console
Router(config)# line vty 0 4
Router(config-line)# authorization exec MYTACAUTH
Router(config-line)# authorization commands 15 MYTACAUTH
```

Authorization is also enabled for all global configuration commands associated with a specific privilege level, as well as commands entered at the console.

Next, the authorization method list is assigned to the vty lines on the router. Users accessing the router through those lines are subject to authorization. If you do not use this command, the default group is used for all lines. Example 8-7 shows that the MYTACAUTH authorization method list is applied to the vty lines for both EXEC and level 15 command access.

Cisco devices also support the capability to use AAA for producing accounting information for user activity. This accounting information can be collected by RADIUS and TACACS+ servers. Again, the RADIUS and TACACS+ servers must already be configured and grouped as part of the authentication configuration.

You must define a method list that provides a sequence of accounting methods by using the following global configuration command:

```
Router(config)# aaa accounting {network | system | exec | commands level} {default
  | list-name} {start-stop | stop-only | wait-start | none} method1 [method2...]
```

The function triggering the accounting can be one of the following keywords:

- **system:** Major router events such as a reload are recorded.
- **exec:** User authentication into an EXEC session is recorded, along with information about the user's address and the time and duration of the session.
- **network:** This applies to network connections, such as PPP.
- **commands:** Information about any command running at a specific privilege level is recorded, along with the user who issued the command.

Next, you can apply an accounting method list to a specific console or vty line on the router. Users accessing the router through that line will have their activity recorded. Use the following line-configuration command to accomplish this:

```
Router(config-line)# accounting {commands level | connection |  exec} {default |
  list-name}
```

If you do not use this command, the default group, if any, will be used for all lines.

In Example 8-8, AAA accounting is configured with a named method list called MYTACACC, using the AAA servers contained in the TACSRVGROUP1 group. User EXEC sessions will be recorded as they start and stop, along with user information. Any commands that are entered while a user is in privilege level 15 (enable mode) will be recorded as well. The MYTACACC method list is applied to the vty lines for EXEC and level 15 commands.

Example 8-8 Configuring AAA Accounting

```
Router(config)# aaa accounting exec MYTACACC start-stop group TACSRVGROUP1
Router(config)# aaa accounting commands 15 MYTACACC start-stop group TACSRVGROUP1
Router(config)# line vty 0 4
Router(config-line)# accounting exec MYTACACC
Router(config-line)# accounting commands 15 MYTACACC
```

Access Control Lists

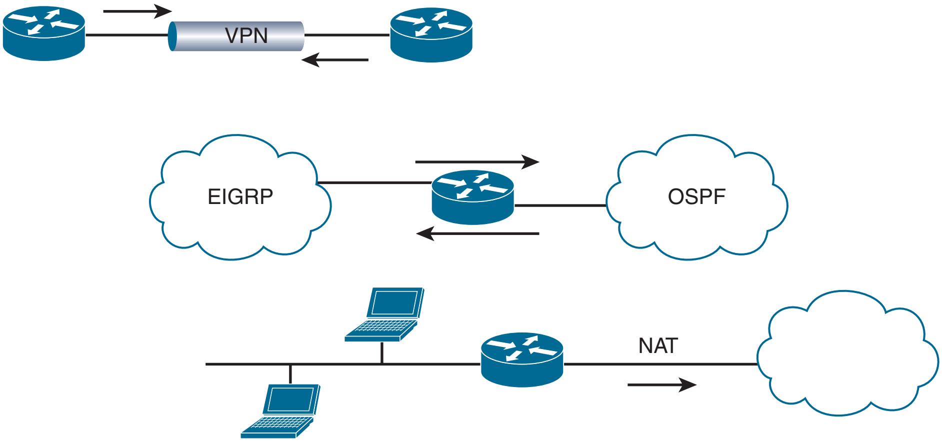
Access control lists (ACLs) are often associated with the control of packets into and out of an interface. However, there are many different applications of ACLs. For example, they can be used to control route advertisements, to specify interesting traffic for a VPN, and to limit debug output. An ACL in isolation is just a packet classification tool. What is done with packets that match the classification depends on how an ACL is applied. The first part of this section focuses on ACLs that control packets traveling into and out of an interface. This section reviews both standard and extended ACLs, using numbered and named configuration methods.

ACL Overview

An ACL is a feature that is used for traffic identification. An ACL enables an administrator to create a set of rules in the form of permit and deny statements that describe which traffic should be identified.

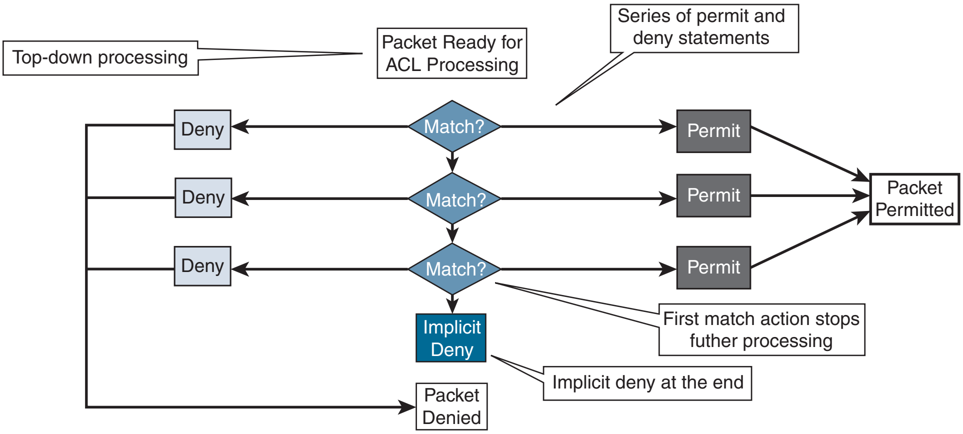
Traffic identification is based on the header values in the IP packet. Identified traffic can receive different treatment, depending on which Cisco IOS function is using the ACLs. For example, ACLs can be used to control route advertisements, to specify interesting traffic for a virtual private network (VPN), and to control which packets trigger Network Address Translation (NAT) address translation, as illustrated in Figure 8-4.

Figure 8-4 ACLs Uses for Traffic Identification



ACLs are supported on a wide range of products, including routers, switches, and firewalls. ACL statements operate in a sequential, logical order. They evaluate packets from top down, one statement at a time. If a packet header and an ACL statement match, the rest of the statements in the list are skipped. The packet is then permitted or denied, as determined by the matched statement. If a packet header does not match an ACL statement, the packet is tested against the next statement in the list. This matching process continues until the end of the list is reached (see Figure 8-5).

Figure 8-5 ACLs Top-Down Processing



A final implied statement covers all packets for which conditions did not test true. This final test condition matches all other packets and results in a deny instruction. The router denies all these remaining packets. This final statement is often referred to as the *implicit deny any* statement. Because of this statement, an ACL should have at least one permit statement in it; otherwise, the ACL denies all packets.

ACL Wildcard Masking

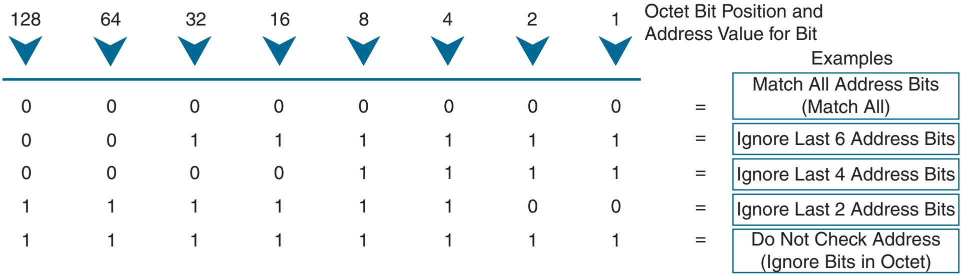
When processing ACLs, a router needs a mechanism to determine which bits of an IP address must match. A wildcard mask describes which bits of an IP address must match in an IP packet to result in a match for a permit or deny statement.

ACL statements include masks, which are also called *wildcard masks*. A wildcard mask is a string of binary digits that tells the router which parts of the subnet number to look at. Although wildcard masks have no functional relationship with subnet masks, they do provide a similar function. The mask determines how much of an IP source or destination address to apply to the address match. The numbers 1 and 0 in the mask identify how to treat the corresponding IP address bits. However, they are used for different purposes and follow different rules. The value 0 says to match the value of the corresponding address bit, while a value of 1 says to ignore the value of the corresponding address bit.

Both wildcard masks and subnet masks are 32 bits long and use binary 1s and 0s. Subnet masks use binary 1s and 0s to identify the network, subnet, and host portion of an IP address. Wildcard masks use binary 1s and 0s to filter individual IP addresses or groups of IP addresses to permit or deny access to resources based on IP address. By carefully setting wildcard masks, you can permit or deny a single IP address or several IP addresses.

Figure 8-6 shows how different wildcard masks filter IP addresses. As you look at the example, remember that binary 0 signifies a match and that binary 1 signifies ignore. A wildcard mask is sometimes referred to as an inverse mask. In a subnet mask, binary 1 is equal to a match, and binary 0 is not a match. The reverse is true for wildcard masks: A 0 in a bit position of the wildcard mask indicates that the corresponding bit in the address must be matched, and a 1 in a bit position of the wildcard mask indicates that the corresponding bit in the address is not interesting and can be ignored.

Figure 8-6 ACLs Wildcard Masking


128 64 32 16 8 4 2 1
Octet Bit Position and Address Value for Bit
Examples
0 0 0 0 0 0 0 0 = Match All Address Bits (Match All)
0 0 1 1 1 1 1 1 = Ignore Last 6 Address Bits
0 0 0 0 1 1 1 1 = Ignore Last 4 Address Bits
1 1 1 1 1 1 0 0 = Ignore Last 2 Address Bits
1 1 1 1 1 1 1 1 = Do Not Check Address (Ignore Bits in Octet)

By carefully setting wildcard masks, you can select a single IP address or many IP addresses. For example, say that you have subnetted a standard Class B address, and you want to create a wildcard mask that matches subnets 172.30.16.0/24 through 172.30.31.0/24. To use one ACL statement to match this range of subnets, use the IP address 172.30.16.0 (the first subnet to be matched) in the ACL, followed by the required wildcard mask. To better understand the process of creating a wildcard mask, look at Figure 8-7 and Table 8-1. The wildcard mask must definitely match the first two octets because the numbers in those two octets are consistent throughout the subnets to be matched. Therefore, the wildcard mask must have all 0s in the first two octets. The wildcard mask must have all 1s in the last octet because it is used for host addresses, and there is no interest in individual hosts. With 1s in the last octet, the wildcard mask ignores the final octet.

In the third octet, where the subnet address occurs, the wildcard mask of decimal 15, or binary 00001111, matches the high-order 4 bits of the IP address. In this case, the wildcard mask matches subnets starting with the 172.30.16.0/24 subnet. For the final (low-end) 4 bits in this octet, the wildcard mask indicates that the bits can be ignored. In these positions, the address value can be binary 0 or binary 1. Therefore, the wildcard mask matches subnets 16, 17, 18, and so on, up to subnet 31. The wildcard mask does not match any other subnets.

Figure 8-7 ACL Wildcard Masking Example

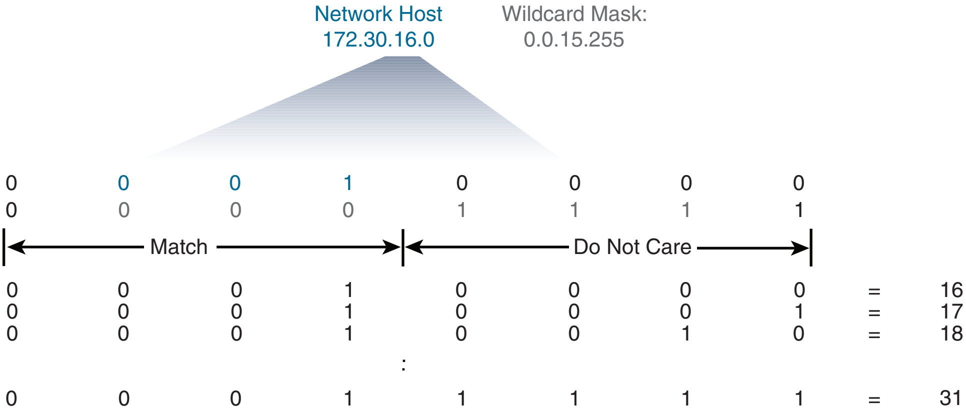

Network Host 172.30.16.0
Wildcard Mask: 0.0.15.255
0 0 0 1 0 0 0 0
0 0 0 0 1 1 1 1
Match
Do Not Care
0 0 0 1 0 0 0 0 = 16
0 0 0 1 0 0 0 1 = 17
0 0 0 1 0 0 1 0 = 18
:
0 0 0 1 1 1 1 1 = 31

Table 8-1 Subnets to Be Matched: 172.30.16.0/24 Through 172.30.31.0/24

	Decimal	Binary
Network address	172.30.16.0	10101100.00011110.00010000.00000000
Subnet mask	255.255.255.0	11111111.11111111.11111111.00000000
Wildcard mask	0.0.?.255	00000000.00000000.????????.11111111

In this example, the address 172.30.16.0 with the wildcard mask 0.0.15.255 matches subnets from 172.30.16.0/24 to 172.30.31.0/24.

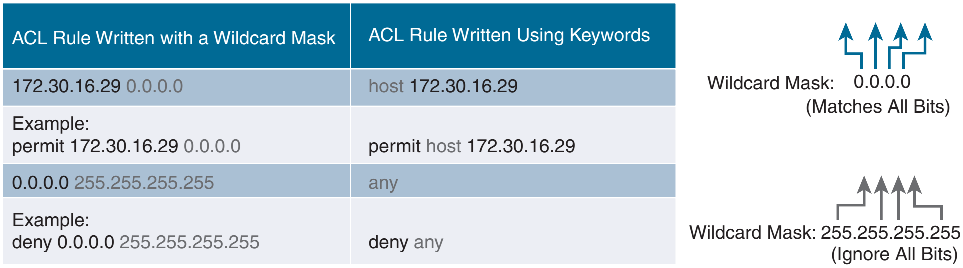
Sometimes, you must use more than one ACL statement to match a range of subnets. For example, to match 10.1.4.0/24 to 10.1.8.0/24, you use 10.1.4.0 0.0.3.255 and 10.1.8.0 0.0.0.255 because 10.1.4.0 with a wildcard mask of 0.03.255 matches the 10.1.4.0/24, 10.1.5.0/24, 10.1.6.0/24, and 10.1.7.0/24 subnets only.

Wildcard Bit Mask Abbreviations

The 0 or 1 bit in an ACL wildcard mask cause the ACL to either match or ignore the corresponding bit in the IP address. Working with decimal representations of binary wildcard mask bits can be tedious. For the most common uses of wildcard masking, you can use abbreviations so that you do not have to enter as many numbers when configuring address test conditions.

In Figure 8-8, instead of entering 172.30.16.29 0.0.0.0, you can use the string **host 172.30.16.29**. Using the abbreviation **host** communicates the same test condition to Cisco IOS Software.

Figure 8-8 ACLs Wildcard Abbreviations



ACL Rule Written with a Wildcard Mask	ACL Rule Written Using Keywords
172.30.16.29 0.0.0.0	host 172.30.16.29
Example: permit 172.30.16.29 0.0.0.0	permit host 172.30.16.29
0.0.0.0 255.255.255.255	any
Example: deny 0.0.0.0 255.255.255.255	deny any

Wildcard Mask: 0.0.0.0
(Matches All Bits)
Wildcard Mask: 255.255.255.255
(Ignore All Bits)

In the second example in Figure 8-8, instead of entering 0.0.0.0 255.255.255.255, you can use the keyword **any** by itself. Using the abbreviation **any** communicates the same test condition to Cisco IOS Software as matching all IP addresses, regardless of subnet mask.

Types of ACLs

There are two main types of ACLs:

- **Standard ACLs:** A standard IP ACL checks the source addresses of the packets that can be routed. Based on the result, the ACL either permits or denies the output for an entire protocol suite, based on the source network, subnet, or host IP address.

- **Extended ACLs:** An extended IP ACL checks both the source and destination packet addresses. This type of ACL can also check for specific protocols, port numbers, and other parameters, which gives administrators more flexibility and control.

You can use two methods to identify standard and extended ACLs:

- **Numbered ACLs:** Numbered standard IPv4 ACLs (1 to 99) test conditions of all IP packets for source addresses. The expanded range is 1300 to 1999. Numbered extended IPv4 ACLs (100 to 199) test conditions of source and destination addresses, specific TCP/IP protocols, and source and destination ports. The expanded range is 2000 to 2699.
- **Named ACLs:** You can identify an ACL by using a descriptive name.

Configuring Numbered Access Lists

Standard IPv4 ACLs, whether numbered (1 to 99 and 1300 to 1999) or named, filter packets that are based on a source address and mask, and they permit or deny the entire TCP/IP protocol suite.

You can configure numbered standard IPv4 ACLs on a Cisco router in global configuration mode. The **access-list** command creates an entry in a standard IPv4 ACL. Example 8-9 shows the syntax of this command.

Example 8-9 Configuring Numbered Standard ACL

```
Router(config)# access-list 1 permit 172.16.0.0 0.0.255.255
Router(config)# access-list 1 permit 192.168.1.0 0.0.0.255
Router(config)# do show access-lists
Standard IP access list 1
      10 permit 172.16.0.0, wildcard bits 0.0.255.255
      20 permit 192.168.1.0, wildcard bits 0.0.0.255

Router(config)# no access-list 1
Router(config)# exit
Router# show access-lists
Router#
```

This ACL example matches any source address that starts with 172.16.x.x and 192.168.1.x. The **show access-lists** command is used to verify the ACL configuration on the router.

To remove the ACL, use the **no access-list** *number* command in global configuration mode. Use the **show access-list** command to confirm that ACL 1 has been removed. With numbered ACLs, you cannot remove individual entries with the **no access-list** command because this command removes the entire ACL. The traditional way of removing or modifying a single numbered ACL entry would be to copy the whole ACL to a text editor, make the changes needed, and remove the entire ACL from the router by using the **no access-list** command. You can then copy the modified ACL from the text editor and paste it in to the router. Newer Cisco IOS Software releases allow easier editing by using sequence numbering. Notice in the output shown in Example 8-9 that the first line of the ACL is numbered with line number 10. It is possible to edit the ACL and remove only line 10 while preserving line 20. Example 8-10 demonstrates this.

Example 8-10 Editing Numbered Standard ACL

```
Router(config)# access-list 1 permit 172.16.0.0 0.0.255.255
Router(config)# access-list 1 permit 192.168.1.0 0.0.0.255
Router(config)# do show access-lists
Standard IP access list 1
      10 permit 172.16.0.0, wildcard bits 0.0.255.255
      20 permit 192.168.1.0, wildcard bits 0.0.0.255

Router(config)# ip access-list standard 1
Router(config-std-nacl)# no 10
Router(config-std-nacl)# do show access-lists
Standard IP access list 1
    20 permit 192.168.1.0, wildcard bits 0.0.0.255
Router(config-std-nacl)#
```

In this example, ACL 1 is reconfigured on the router. The **ip access-list standard** command is then used to enter standard named access list configuration mode. The **no 10** command deletes line 10 while preserving line 20.

Configuring Numbered Extended IPv4 ACLs

A standard ACL can specify only source IP addresses and source networks, so it is not possible to filter to a specific destination. For more precise traffic filtering, you should use extended ACLs.

Extended ACLs provide a greater range of control than do standard ACLs. In addition to verifying packet source addresses, extended ACLs also check destination addresses, protocols, and port numbers. They provide more criteria on which to base the ACL. For example, an extended ACL can simultaneously allow email traffic from a network to a specific destination and deny file transfers and web browsing for a specific host.

The ability to filter on a protocol and port number allows you to build very specific extended ACLs. Using the appropriate port number, you can specify an application by configuring either the port number or the name of a well-known port.

You can configure numbered extended IPv4 ACLs on a Cisco router in global configuration mode. The **access-list** command is used to create an entry in an extended IPv4 filter list because a number between 100 and 199 is used to identify the ACL. Example 8-11 shows an extended ACL being configured on the router. The first line of the ACL denies FTP traffic from subnet 172.16.4.0/24 to subnet 172.16.3.0/24, and the second line of the ACL permits all other traffic.

Example 8-11 Configuring Numbered Extended ACL

```
Router(config)# access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0
  0.0.0.255 eq 21
Router(config)# access-list 101 permit ip any any
Router(config)# do show access-lists 101
Extended IP access list 101
    10 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq ftp
    20 permit ip any any
```

For extended access lists, you can specify either the name or the number of the protocol. The most commonly used keywords are **ip**, **tcp**, **udp**, and **icmp**.

When TCP or UDP is specified as the protocol, it is possible to include **lt** (less than), **gt** (greater than), **eq** (equal), **neq** (not equal), and **range** (inclusive range) to match specific network applications by name (for example, www, FTP, SSH) or number (for example, 80, 21, 22).

The output of the **show access-lists 101** command displays only the newly added extended ACL that was configured on the router.

Notice that the port number 21 that was initially configured was automatically replaced by the keyword **FTP**.

As with a standard access list, using the **no access-list 101** command removes the entire extended ACL 101 from the router. To delete or add entries to the ACL, as shown earlier for standard ACLs, you can use the **ip access-list extended** command.

Configuring Named Standard ACLs

Naming an ACL makes it easier to understand the function of the ACL. For example, an ACL to deny one subnet could be called NO_Subnet. When you identify an ACL with a name instead of a number, the configuration mode and command syntax are slightly different. The alphanumeric name string must be unique. If sequence numbers are not configured, they are generated automatically, starting at 10 and incrementing by 10.

You use the access list configuration mode to define named ACLs. To enter this mode, use the **ip access-list** command, as illustrated in Example 8-12.

Example 8-12 Configuring a Named Standard ACL

```
Router(config)# ip access-list standard SERVER_ACCESS
Router(config-std-nacl)# deny 10.1.1.0 0.0.0.255
Router(config-std-nacl)# permit any
Router(config-std-nacl)# exit

Router# show access-lists
Standard IP access list Subnet_ONLY
     10 deny 10.1.1.0 0.0.0.255
     20 permit any
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# ip access-list standard SERVER_ACCESS
Router(config-std-nacl)# 5 permit host 10.1.1.25
Router(config-std-nacl)# end
Router# show access-lists
Standard IP access list SERVER_ACCESS
     5 permit host 10.1.1.25
    10 deny 10.1.1.0 0.0.0.255
    20 permit any
```

The output in Example 8-12 shows the commands that are used to configure a standard ACL named SERVER_ACCESS on the router. The ACL initially denies traffic from hosts on the 10.1.1.0/24 subnet while permitting all other traffic. Capitalizing ACL names is not required, but it makes them stand out when you view the running configuration output.

Named IP ACLs also allow you to add, modify, or delete individual entries in a specific ACL. You can use sequence numbers to insert statements anywhere in a named ACL.

When you add statements to an ACL, the default increment is 10. Example 8-12 shows an extra entry that is numbered 5 in the SERVER_ACCESS ACL; it is inserted in front of line 10 to permit a specific host.

Note that a reload changes the sequence numbers in an ACL. After a reload of the ACL in Example 8-12, the sequence numbers will be 10, 20, 30 instead of 5, 10, 20. You can use the **access-list resequence** command to renumber the ACL entries in an ACL without having to reload.

Configuring Named Extended ACLs

Named extended ACLs are created in essentially the same way as named standard ACLs. The ACL configuration mode is also used to configure a named extended ACL.

Example 8-13 shows the commands that are used to configure an extended ACL named WWW-ACCESS on the router. The ACL permits TCP port 80 and port 443 traffic from hosts on the 10.1.1.0/24 subnet to any destination. Notice that the CLI automatically replaces the numerical port number (80) to display the www keyword instead. Also notice that line numbers 30 and 40 are used here. This is strictly a demonstration of the options available when configuring a named ACL.

Example 8-13 Configuring a Named Extended ACL

```
Router(config)# ip access-list extended WWW-ACCESS
Router(config-ext-nacl)# 30 permit tcp 10.1.1.0 0.0.0.255 any eq 80
Router(config-ext-nacl)# 40 permit tcp 10.1.1.0 0.0.0.255 any eq 443
Router(config-ext-nacl)# do show access-lists WWW-ACCESS
Extended IP access list WWW-ACCESS
    30 permit tcp 10.1.1.0 0.0.0.255 any eq www
    40 permit tcp 10.1.1.0 0.0.0.255 any eq 443
```

Applying ACLs to Interfaces

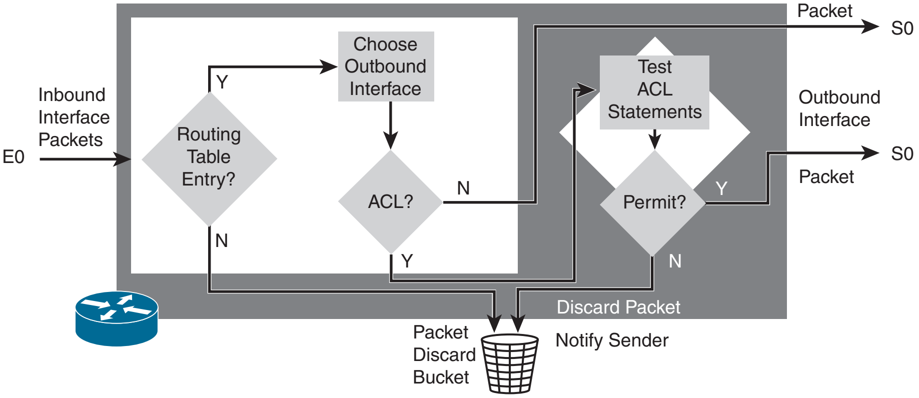
When you use ACLs for traffic filtering, they can operate in the inbound direction or in the outbound direction, from the perspective of the device being configured. The direction determines at which point packets are tested against the ACL as they pass through the device (router, switch, firewall):

- **Outbound ACLs:** Incoming packets are routed to the outbound interface and then are processed through the outbound ACL. If packets match a permit statement, they are forwarded through the interface. If packets match a deny statement or if there is no match, they are discarded.
- **Inbound ACLs:** Incoming packets are processed by the ACL before they are routed to an outbound interface. An inbound ACL is efficient because it eliminates the overhead of routing

lookups if the filtering tests deny the packet and it is discarded. If the tests permit the packet, the packet is processed for routing.

Figure 8-9 illustrates the process used when a packet arrives inbound on a router's interface and an ACL has been applied outbound on the exit interface.

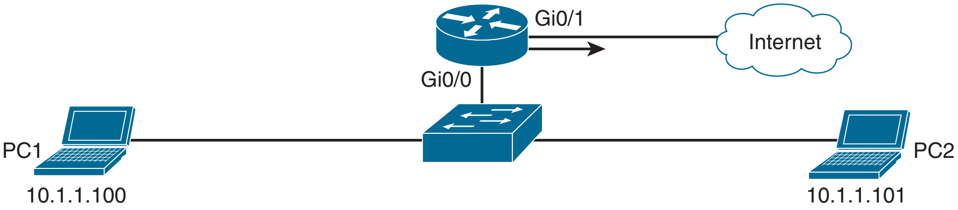
Figure 8-9 ACL Outbound Filtering


Inbound
Interface
Packets
E0
Routing
Table
Entry?
Y
N
Choose
Outbound
Interface
ACL?
N
Y
Test
ACL
Statements
Permit?
Y
N
Packet
S0
Outbound
Interface
S0
Packet
Discard Packet
Notify Sender
Packet
Discard
Bucket

After you have configured an ACL for packet filtering, you need to apply the ACL to an interface by using the **ip access-group** command. Only one ACL per protocol, per direction, and per interface is allowed.

Figure 8-10 shows the topology used for this configuration example.

Figure 8-10 Applying ACLs to Router Interfaces


Gi0/1
Internet
Gi0/0
PC1
10.1.1.100
PC2
10.1.1.101

Example 8-14 shows a scenario in which ACL 1 is applied in the *outbound* direction on the branch router GigabitEthernet0/1 interface to provide traffic filtering. ACL 1 includes a deny statement that matches traffic from a specific host with IP address 10.1.1.101. The second line in the ACL permits traffic from hosts in the 10.1.1.0 /24 network. It is important to specify a permit statement because an ACL ends with an implicit deny all statement.

Example 8-14 Applying a Standard ACL

```
Router(config)# access-list 1 deny host 10.1.1.101
Router(config)# access-list 1 permit 10.1.1.0 0.0.0.255
Router(config)# interface GigabitEthernet 0/1
Router(config-if)# ip access-group 1 out
```

Example 8-15 shows a scenario in which ACL 101 is applied in the *inbound* direction on the branch router GigabitEthernet0/0 interface to provide traffic filtering similar to that of the first example. ACL 101 includes a deny statement that matches IP traffic from host 10.1.1.101 to any destination. The second line in the ACL permits IP traffic from hosts in the 10.1.1.0/24 network to any destination.

Example 8-15 Applying an Extended ACL

```
Router(config)# access-list 101 deny ip host 10.1.1.101 any
Router(config)# access-list 101 permit ip 10.1.1.0 0.0.0.255 any
Router(config)# interface GigabitEthernet 0/0
Router(config-if)# ip access-group 101 in
```

Examples 8-14 and 8-15 achieve the same result: PC2 is denied Internet access. The advantage of using the inbound extended ACL in Example 8-15 is that incoming packets are processed as they enter the interface, before they are routed to the outbound interface. An inbound ACL is efficient because it saves the overhead of routing lookups if the packet is discarded. If the packet is permitted by the ACL, it is then processed for routing by the router.

To remove an ACL from an interface, enter the **no ip access-group** command on the interface and then enter the global **no access-list** command to remove the entire ACL, if needed.

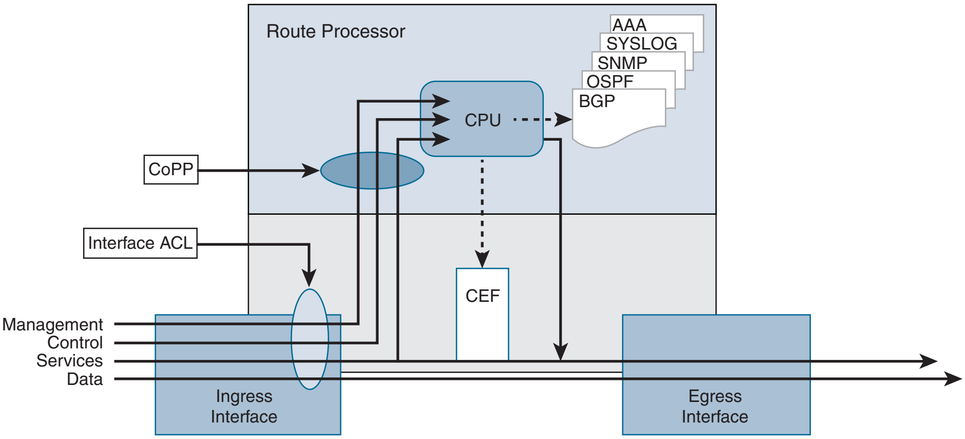
Control Plane Policing

Many Cisco IOS Software security features are available to protect the route processors of networking devices. Some of these features are generic and applicable to a broad range of security functions, such as interface ACLs. Others, such as control plane policing (CoPP), are specifically designed to protect the route processor.

Using an interface ACL is the traditional and most generally available approach for managing all packets entering or exiting a network device. These ACLs are well understood and are generally applicable to data, services, control, and management plane packets. However, as illustrated in Figure 8-11, ACLs are applied at the interface level to each packet ingressing (or egressing) the interface—not just control plane packets. In addition, an ACL must be applied to every individual interface to which the policy is to be applied. On large routers, this can be an onerous task.

CoPP is a Cisco IOS feature that is designed to allow administrators to specify controls over traffic that is directed to a device's control plane. The goal is to prevent low-priority or unnecessary traffic from overwhelming system resources, which could lead to issues in system performance. CoPP treats the control plane as a separate entity with its own ingress and egress ports. CoPP facilitates the definition of rules to control traffic traversing the control plane's ingress and egress ports, as illustrated in Figure 8-11. CoPP allows you to specify what traffic is allowed and what traffic is denied on the control plane interfaces, and it also provides for rate limiting of allowed traffic. Therefore, expected traffic that is of lower priority can be allowed, but attacks where floods of lower-priority traffic are sent to the control plane can be mitigated.

Figure 8-11 Protecting the Control Plane



CoPP allows the implementation of QoS features for the control plane. As such, it is configured with the IOS Modular QoS CLI (MQC) framework. MQC involves three main concepts:

- **Class maps:** Class maps define characteristics that are used to map traffic to classes. That is, class maps categorize traffic into classes. There are several mechanisms available within class maps, but the most common use is to reference an ACL. If traffic is permitted by the referenced ACL, then traffic belongs to the associated class. When implementing CoPP, you might define class maps with names such as Routing, Management, Normal, or Undesirable.
- **Policy maps:** Policy maps define actions that should be taken on particular classes of traffic. When implementing CoPP, police statements are used in policy maps to specify traffic flow rates and burst rates, along with policy on what to do with conforming and nonconforming traffic.
- **Service policies:** To specify where a policy map should be implemented, you use a service policy statement. When implementing CoPP, control plane configuration mode is reached by using the **control-plane** command in global configuration mode. The **service-policy** command is then applied in control plane configuration mode.

CoPP Configuration

CoPP is configured almost the same way as quality of service (QoS) on a physical interface; the difference is that for CoPP, the service policy is applied to the control plane. MQC is used to configure QoS, and this process can be divided into three steps:

1. **Defining the traffic:** In the first step, the interesting traffic is defined in a class map. A common method of defining interesting traffic is to create an access list and reference it in a class map.
2. **Defining a service policy:** In this step, the QoS policy and associated actions are defined in a policy map. For CoPP, the **police** option is used to control how much traffic is sent to the control plane.
3. **Applying the service policy:** The last step is to apply the service policy to the correct interface. Normally, a QoS policy is applied to a physical interface, but in the case of CoPP, it is applied in control plane configuration mode.

Example 8-16 shows an entire CoPP configuration example.

Example 8-16 Configuring CoPP

```
Router(config)# access-list 100 permit udp any any eq snmp
Router(config)# access-list 100 permit tcp any any eq www
Router(config)# access-list 100 permit tcp any any eq 443
Router(config)# access-list 100 permit tcp any any eq 22
Router(config)# class-map COPP_class
Router(config-cmap)# match access-group 100
Router(config-cmap)# exit
Router(config)# policy-map COPP_policy
Router(config-pmap)# class COPP_class
Router(config-pmap-c)# police 300000 conform-action transmit exceed-action drop
Router(config-pmap-c)# class class-default
Router(config-pmap-c)# police rate 50 pps conform-action transmit exceed-action
  dropf
Router(config-pmap-c)# exit
Router(config)# control-plane
Router(config-cp)# service-policy input COPP_policy
```

This example creates a class map for all SNMP, HTTP, HTTPS, and SSH traffic that was matched using ACL 100. The example also shows a policing policy applied to the CoPP class map. The QoS policy is configured to police all SNMP, HTTP, HTTPS, and SSH traffic to 300 Kbps and to drop any traffic that exceeds this rate. A second class, **class-default**, is rate limited to 50 packets per second. The class **class-default** is special in MQC because it is always automatically placed at the end of every policy map. Match criteria cannot be configured for **class-default** because it automatically includes an implied match for all packets. Finally, the CoPP policy is applied to the control plane in the **input** direction for packets sent to the route processor.

To verify the CoPP policy, you can use the following commands:

- **show access-list:** This command displays all configured ACLs on the router and any hit counters associated with any ACL entries that have seen packet matches. When you include the number or name of a specific ACL, only that ACL will be displayed.
- **show class-map:** This command displays all configured class maps on the router. When you include the name of a specific class map, only the specific policy map will be displayed.
- **show policy-map:** This command displays all configured policy maps on the router. When you include the name of a specific policy map, only the specific policy map will be displayed.
- **show policy-map control-plane:** This command displays dynamic information about the policy applied, including rate information and the number of bytes (and packets) that conformed to or exceeded the configured policies. In other words, it presents only the output relevant to the CoPP configuration.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	**Module, Chapter, or Link**
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	26
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	9, 10

Day 7

Virtualization

ENCOR 350-401 Exam Topics

Virtualization

- Describe device virtualization technologies
  - Hypervisor type 1 and 2
  - Virtual machine
  - Virtual switching
- Configure and verify data path virtualization technologies
  - VRF

Key Topics

Today we review topics related to *virtualization*—the process of running one or more virtual instances of an operating system in a layer abstracted from the underlying hardware. Virtualization commonly refers to simultaneously running multiple operating systems on general- or specific-purpose hardware. To the applications running on top of the virtualized machine, it may appear as if the applications are on their own dedicated machines, where the operating system, libraries, and other applications are unique to the (guest) virtualized system. Virtualization in networks allows for a single physical network topology to be utilized by different networks that have no interaction with each other. Today we explore server virtualization and network functions virtualization, as well as virtual routing and forwarding (VRF).

Server Virtualization

The heart of the virtualized server environment is the virtualization software. In this section, we explore virtualization and the VMware ESXi (Elastic Sky X integrated) software.

Standard deployment of applications on computer systems involves installing a single operating system on each computer. To provide the greatest stability in business environments, good practice dictates that a single type of application be present on an operating system; otherwise, compatibility issues could cause unforeseen problems. This approach provides good stability but is not very cost-effective. A growing number of applications greatly expands the number of physical servers in a data center, which increases the total cost of ownership (TCO) and can adversely impact the IT budget of a company.

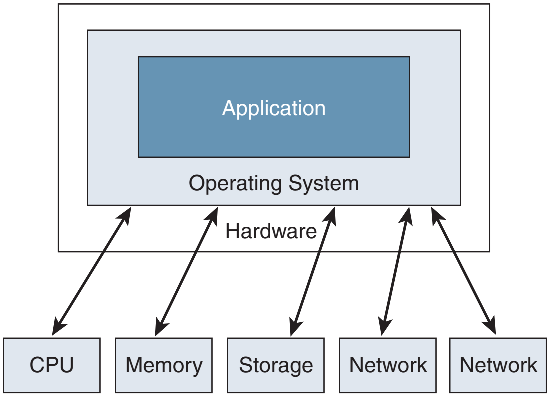
This standard deployment method of server usage was common in unified communications applications such as Communications Manager 7.x. The installation media installed the Voice Technology Group (VTG) software (a hardened version of Linux) on the bare-metal Media Convergence Server (MCS) 7800 series server and deployed a unified communications application such as Cisco Unified Communications Manager or Cisco Unity Connection on the VTG operating system. Each physical server represented one node in the application cluster.

Physical Server

Virtualization, in the context of IT, helps abstract physical resources from the services they provide. In server virtualization, a layer of software is installed between the server hardware and the operating system.

In a non-virtualized server, the operating system (Windows or Linux) has direct access to the physical resources on the server, as shown in Figure 7-1. If the physical server has four CPU cores, all four cores are available for use by the operating system.

Figure 7-1 Physical Server Resources

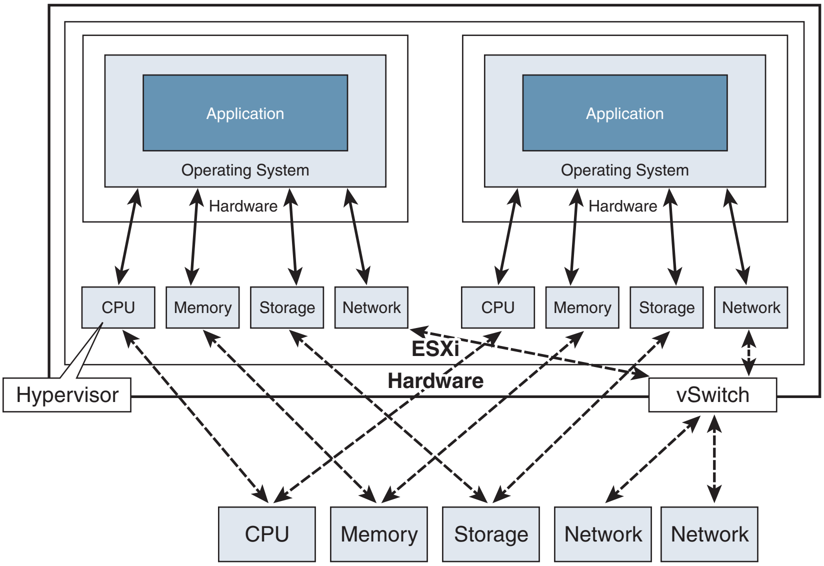


Virtualized Server

With a virtualized server, instead of loading an operating system, such as Linux, on the server, a software package called a hypervisor is installed on the server. The hypervisor is a combination of a lightweight operating system and additional software to create and manage the virtual environment, which abstracts the underlying hardware compute, storage, and network from the virtual servers running atop the hypervisor. An example of hypervisor software is VMware ESXi (vSphere).

Virtual machines (VMs) are created with software representations of CPU cores, memory, hard disk space, and network interface cards (vNICs). The application operating system is not aware that the resources are virtualized, and the resources that the VM has access to are not the physical CPUs or memory, as shown in Figure 7-2.

Figure 7-2 Virtualized Server Resources



The operating system has access to the assigned resources in the normal way. When the hypervisor receives a request, it forwards the request to the physical resources of the host server.

In each case, the virtual resource is a portion of the physical resources of the host server. One exception is the vNIC of the VM. Rather than connect directly to the physical Ethernet ports of the host server, the vNIC connects to an internal virtual Ethernet switch called a vSwitch via vPorts. The vSwitch can have many connections from VMs. The vSwitch maintains connections, called uplinks, to the physical Ethernet interfaces.

This approach has several benefits, including the following:

- The virtualization layer provides a uniform virtualized hardware interface to the operating system that is installed on top. Even if three different physical servers in the data center have been provided by three different vendors, they appear the same to the operating system that is installed on top of the virtualization layer. There may be differences in performance, but there are no differences in the type of resources, such as the network cards or graphics adapters, that appear to the operating system.
- The virtualization layer is capable of segmenting the physical hardware into multiple separate resource units that all draw from the same physical pool. This means you can have multiple instances of a single operating system or even different types of operating systems running simultaneously on a single server. These separate instances are all completely independent of each other. Unless a non-redundant hardware component fails, these instances are independent of each other, as if they were running on separate physical servers.

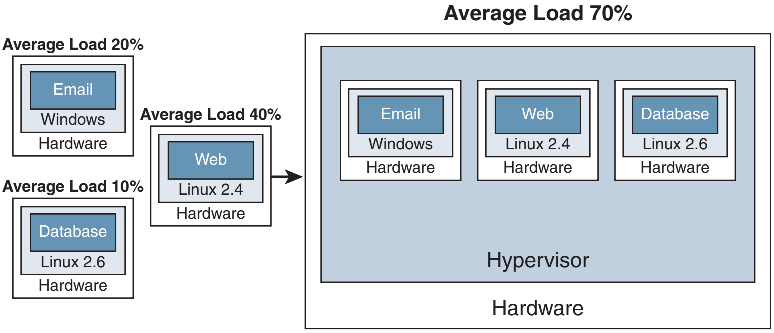
Basic Virtualized Server Environment

Server virtualization gives you the ability to run multiple operating systems on a single physical server. This ability can be useful in testing or production environments.

For example, a production virtual operating system can be copied and used to test upgrade procedures in a non-production or test environment. You can therefore try an upgrade procedure in a virtual environment that is the same as the production environment used by employees, but mistakes or compatibility problems will not affect users. After the upgrade has been thoroughly tested and the consequences evaluated, a real upgrade can take place, based on a full understanding of the impact on the production environment.

Physical servers are often underutilized in non-virtualized environments because typically a separate server is assigned for each application, as illustrated in Figure 7-3. An application such as Cisco Unified Communications Manager (CUCM), with low numbers of registered devices or applications, might use less than 20% CPU when operating normally. That leaves 80% of the system resources unused. Underutilization has a direct impact on both operations and operating costs as it increases the number of servers required. Every extra server requires additional physical space, power, and cooling requirements. As the number of servers grows, it becomes more difficult to manage them all.

Figure 7-3 Example of Underutilized Server Resources



Depending on the average load of an existing deployment, with virtualization, it is not unusual to be able to put 12 or more servers onto a single piece of hardware. If the data center environment is designed with virtualization in mind, the ratio of virtual servers to physical servers can be much higher. The limiting factor with collaboration applications, as well as other applications, is that you cannot exceed the physical resources of the host server.

For example, if a host server has 24 CPU cores installed (that is, 2 12-core processors), you could add 24 VMs that need 1 vCPU each. However, if a video conferencing application, such as Cisco Meeting Server (CMS), needs 20 vCPUs and other applications require 2 vCPUs each, you would be able to have only 3 VMs on that host.

Virtualization also provides more flexibility as the deployment of a new virtual server is much easier and faster than the deployment of a physical server.

Hypervisor: Abstraction Layer

A hypervisor acts as an abstraction layer that sits between the guest operating system and the hardware. A hypervisor, such as VMware ESXi, is a dedicated operating system that leverages VMkernel for separation of the virtual and physical layers. The abstraction layer supports the installation of multiple operating systems, which can be of different types, but a hypervisor supports only virtual machines and is not intended for the installation of applications. The component of the hypervisor that conveys commands from the software on the VM to the actual hardware on the physical server is known as the virtual machine monitor.

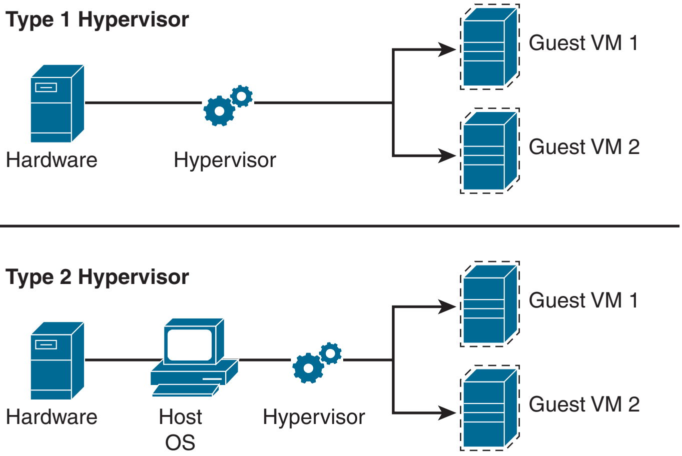
Cisco unified communications applications started supporting virtualization with VMware ESXi 4.0 and Cisco UCM Version 8.0. Most current versions of Cisco collaboration applications support ESXi Version 5.0 and later.

A hypervisor performs several tasks:

- Provides provisioned resources to individual guest operating systems or VMs by partitioning the resources of the physical server, or host, on which it is installed
- Provides connectivity between VMs and other VMs, as well as external network resources or physical environment
- Ensures separation between individual VMs from a security perspective

Two types of hypervisors are used in networks today: type 1 and type 2 (see Figure 7-4).

Figure 7-4 Type 1 and Type 2 Hypervisors



Type 1 Hypervisors

A type 1 hypervisor runs directly on the host machine's physical hardware and does not have to load an underlying OS first. This type of hypervisor is often referred to as a *bare-metal hypervisor*. With direct access to the underlying hardware and no other software, such as operating systems and device drivers, to contend with, type 1 hypervisors are regarded as the most efficient and best-performing hypervisors available for enterprise computing. Hypervisors such as VMware ESXi, Microsoft Hyper-V Server, and open-source KVM are examples of type 1 hypervisors.

Type 2 Hypervisors

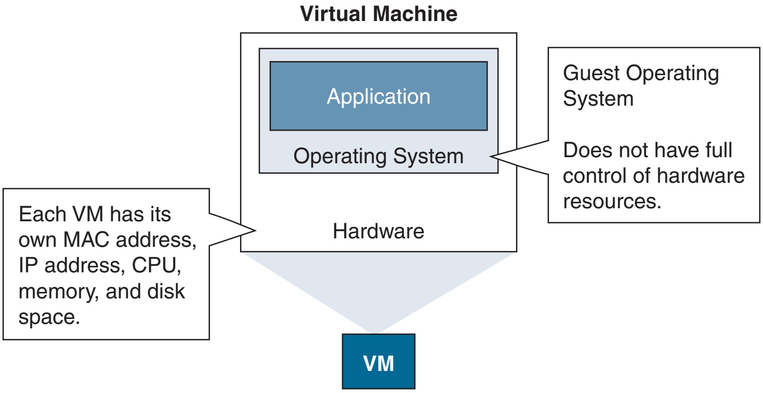
A type 2 hypervisor is typically installed on top of an existing OS and is called a *hosted hypervisor* because it relies on the host machine's preexisting OS to manage calls to CPU, memory, storage, and network resources. Type 2 hypervisors include VMware Fusion, Oracle VM VirtualBox, Oracle VM Server for x86, Oracle Solaris Zones, Parallels, and VMware Workstation. This type of nested deployment is usually avoided in production environments.

VM Definition

A *virtual machine* (*VM*) is a logical container that holds all the resources that an operating system requires for normal operation, such as a graphics adapter, memory, a processor, and networking resources. As far as the operating system running in a VM is concerned, there is no difference between these components and the components that would be available on a physical server. However, these components are virtualized representations of host resources. In the case of processor capacity or memory, the assigned resources are a percentage of the actual resources. In the case of a hard disk, the virtual component is a specially formatted file that is visible as a disk to the virtualized (guest) operating system. Regarding the vNIC, this is a simulated virtual component that is managed by the hypervisor, so it acts like a physical component.

A VM is the virtual equivalent of a physical PC or server, and it requires the same set of software and networking identifiers as any physical device, as shown in Figure 7-5. One of the benefits of a VM is that an administrator can set these identifiers to desired values by simply manipulating the VM configuration file.

Figure 7-5 Simple Virtual Machine Architecture Example


Virtual Machine
Application
Operating System
Hardware
VM
Guest Operating System
Does not have full control of hardware resources.
Each VM has its own MAC address, IP address, CPU, memory, and disk space.

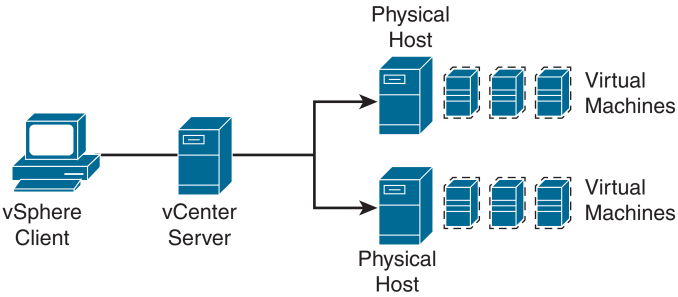
Managing Virtual Machines

There are two ways of managing virtual machines when using a solution like VMware vSphere. An ESXi host can be controlled and configured either individually or as part of a group of hosts that are centrally controlled via the vSphere GUI.

Individual access is possible by using the vSphere Client for Windows or the newer VMware Host Client. The more mature access method is the Windows client. The client is downloaded from the VMware website and installed on the client PC. The same client can connect to an individual VMware host or to VMware vCenter Server.

Typically, a larger VMware environment is managed using VMware vCenter Server (see Figure 7-6). vCenter is a unified control point for all the participating hosts. You can configure hosts on an individual basis in simpler environments, but some advanced functionalities of VMware, such as vMotion (which enables you to easily move a VM from one host to another), require vCenter.

Figure 7-6 Centralized vSphere Management



In addition to being able to configure multiple hosts from one interface, you can activate additional features, such as vSphere Distributed Switch (vDS), resource and power load balancing, and CPU compatibility. vDS provides a centralized interface for configuration, administration, and monitoring of virtual machine access and switching in a data center.

vCenter can be deployed as a separate physical server, or it can be installed as a VM. Starting in vSphere 5.0, vCenter is available as a Linux-based virtual appliance that can be downloaded and installed from a virtual appliance template.

Network Function Virtualization

Enterprise branch architectures today often host multiple devices for accomplishing required network functionality. Network appliances that perform WAN optimization, firewall protection, and Cisco intrusion prevention system (IPS) or voice services operate alongside routers and switches. In high-end branch environments, these devices are even deployed with redundancy, further adding to the hardware landscape that has to be installed and operated in the branch.

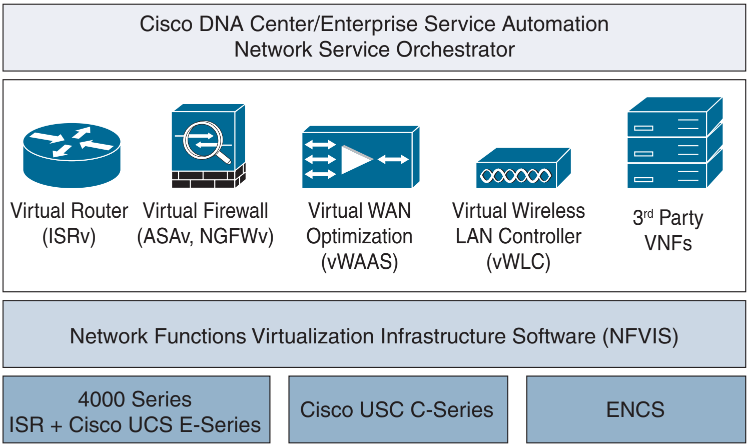
Cisco Enterprise Network Functions Virtualization (Enterprise NFV) reduces the operational complexity of such branch environments by running the required networking functions as software on standard x86-based hosts. In particular, the Cisco Enterprise NFV solution reduces the number of hardware elements to be managed at the branch, reducing the need to perform costly site visits for hardware installations or upgrades. Also, the Cisco Enterprise NFV solution automates the deployment, management, and operations of branch functions, which diminishes operating expenses.

Cisco Enterprise NFV Solution Architecture

Cisco Enterprise NFV is an end-to-end solution that addresses all requirements for deploying virtualized network and application services, such as orchestration and management, virtualization software packages, and options for different hardware platforms. Cisco Enterprise NFV also addresses the use of Cisco and third-party service functions.

Figure 7-7 illustrates the Cisco Enterprise NFV solution architecture.

Figure 7-7 NFV Solution Architecture


Cisco DNA Center/Enterprise Service Automation
Network Service Orchestrator
Virtual Router (ISRv)
Virtual Firewall (ASAv, NGFWv)
Virtual WAN Optimization (vWAAS)
Virtual Wireless LAN Controller (vWLC)
3rd Party VNFs
Network Functions Virtualization Infrastructure Software (NFVIS)
4000 Series ISR + Cisco UCS E-Series
Cisco USC C-Series
ENCS

The top layer of the Cisco Enterprise NFV architecture is focused on orchestration, automation, and management (OAM). This layer is composed of either Cisco Enterprise Service Automation (ESA) or Cisco DNA Center for network service automation and orchestration. It is also possible to use Cisco Network Services Orchestrator (NSO) for OAM in NFV environments. This layer provides common OAM across both virtual and physical devices.

The second layer is composed in the virtual network functions (VNFs). VNFs are the service functions that execute the desired tasks, such as Layer 3 routing, firewall and intrusion detection and prevention systems, WAN optimization and caching, SD-WAN vEdge and cEdge routers, and virtual cloud wireless LAN controllers. Also, third-party or custom VNFs can be deployed at this layer (for example, Palo Alto, Fortinet, ThousandEyes).

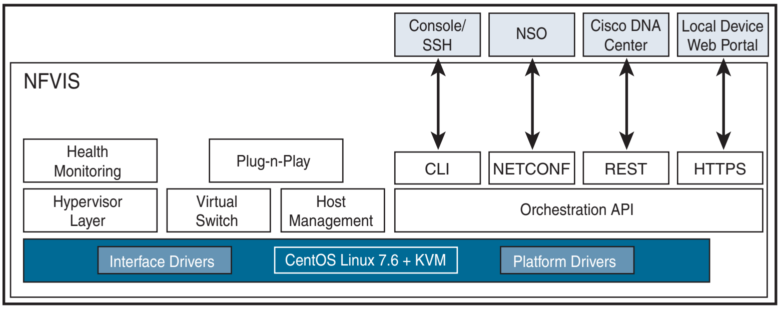
The third layer is Cisco Enterprise NFV Infrastructure Software (NFVIS). NFVIS is a software host that manages virtualization and hardware. An integrated hypervisor allows you to create and run network functions as virtual appliances using a GUI. Also, it manages the underlying hardware platform and exposes an API for orchestration and management.

The bottom layer is the platform hardware layer. It is composed of various host options for different branch sizes. The supported platforms are the Cisco Unified Computing System (UCS) C-Series and the Cisco Integrate Services Router (ISR) 4000 routers, which are equipped with Cisco UCS E-Series servers and Cisco Enterprise Network Compute System (ENCS). Cisco ENCS is a compute appliance that is designed for a virtualized software-defined network architecture. The system offers service flexibility and performance, and it lowers total cost of ownership for the next-generation branch office.

NFVIS Building Blocks

The Cisco Enterprise NFV solution includes the NFVIS virtualized software platform. NFVIS extends Linux by packaging additional functions for VNF lifecycle management, monitoring, device programmability, and hardware acceleration. Figure 7-8 shows the building blocks of NFVIS.

Figure 7-8 NFVIS Building Blocks


Console/ SSH
NSO
Cisco DNA Center
Local Device Web Portal
NFVIS
Health Monitoring
Plug-n-Play
CLI
NETCONF
REST
HTTPS
Hypervisor Layer
Virtual Switch
Host Management
Orchestration API
Interface Drivers
CentOS Linux 7.6 + KVM
Platform Drivers

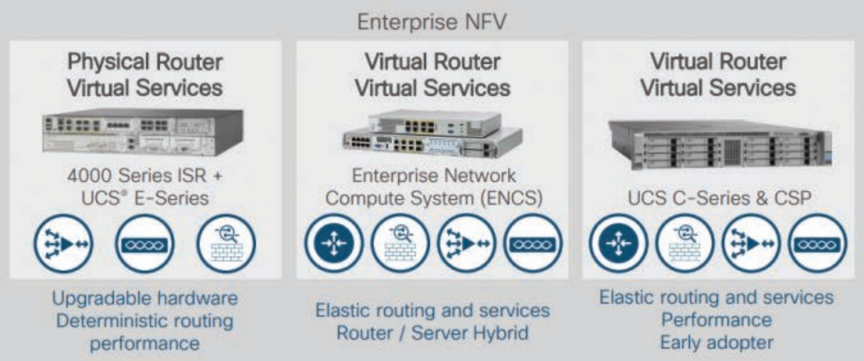
NFVIS delivers the following components and functionality:

- **Linux (CentOS Version 7.1) and OS kernel:** NFVIS is built on CentOS Version 7.1. The OS kernel component drives the underlying hardware platforms (for example, Cisco UCS C-Series servers, Cisco UCS E-Series servers, Cisco ENCS) and hosts the virtualization layer for VNFs, virtual switching APIs, and management.
- **Virtualization support and virtual switching:** The hypervisor for virtualization is built on top of a KVM, and it includes QEMU, libvirt, and other associated processes. Virtual switching enables multiple VNFs to share the physical interface resources and allows the traffic to be passed within the x86 host, between VNFs.
- **VM lifecycle management:** VM lifecycle management includes support to bring up VNFs dynamically and to control their liveliness using ESC-lite. Cisco Elastic Services Controller (ESC) is a VNF manager that performs lifecycle management. Cisco ESC provides agentless and multivendor VNF management by provisioning virtual services and monitors the health and load of these services. Cisco ESC provides the flexibility to define rules for monitoring and associates the actions to be triggered based on the outcomes of these rules.
- **Orchestration API:** Cisco Enterprise NFVIS provides orchestration APIs for OAM servers to automate and orchestrate the VNFs on NFVIS. REST, CLI, or NETCONF, and YANG are supported.
- **PnP client:** The PnP agent that is running on the NFVIS host first tries to discover a PnP server that is running on the Cisco DNA Center with which it can communicate. Cisco DNA Center is loaded with the right host configuration. When the PnP server is found and the connection is established, the agent performs deployment-related activities such as host configuration with the server.
- **Web server:** The web server can enable connectivity to NFVIS through HTTPS, which is particularly useful for supporting local management tools and orchestration APIs.
- **Health monitor:** Tools like syslog, SNMP, and collectd help with statistics collection and reporting.

Cisco NFV Hardware Options

The compute resources to operate VNFs are offered in the Cisco Enterprise NFV solution in various form factors, as shown in Figure 7-9:

Figure 7-9 NFV Hardware Options


Enterprise NFV
Physical Router Virtual Services
4000 Series ISR + UCS® E-Series
Upgradable hardware
Deterministic routing performance
Virtual Router Virtual Services
Enterprise Network Compute System (ENCS)
Elastic routing and services
Router / Server Hybrid
Virtual Router Virtual Services
UCS C-Series & CSP
Elastic routing and services
Performance
Early adopter

- **Cisco UCS C-Series servers:** These servers can integrate into Cisco UCS through Cisco UCS fabric interconnects or can be used as standalone servers with Cisco or third-party switches. With ultrafast performance for two- and four-socket servers, Cisco rack and storage servers can integrate into Cisco UCS through a single set of cables. These servers provide a wide range of I/O, memory, internal disk, solid-state disk (SSD) drive, and NVMe storage device capacity, enabling you to easily match servers to workloads. The Cisco UCS C480 ML M5 server is designed for artificial intelligence and machine learning workloads. Its purpose-specific design incorporates eight NVIDIA V100 SMX2 32 GB graphics processing units (GPUs) to power compute-intensive deep learning applications.
- **Cisco ISR 4000 routers and Cisco E-Series servers:** Cisco 4000 Series ISR devices are branch office edge routers that are designed to meet Gigabit forwarding performance needs while running a broad set of network and application services on a single platform. The modularity of this series, with available interface modules, transforms the network edge at the branch into a premier NFV host. When equipped with Cisco UCS E-Series compute servers running NFVIS, the result is the most optimally deployed NFV services available in the market. This flexible platform is further extended using network interface modules to add a wide range of options for network interfaces, voice for unified communications, storage, and other x86 compute cards.
- **Cisco ENCS:** The Cisco 5000 Enterprise Network Compute System (ENCS) is a line of compute appliances designed for the Cisco SD-Branch and Enterprise NFV solution. It delivers a new standard of software-defined flexibility and performance and offers a low total cost of ownership (TCO). The 5000 ENCS is a hybrid platform that combines the best attributes of a traditional router and a traditional server and offers the same functionality with a smaller infrastructure footprint. Offered with the Cisco Integrated Services Virtual Router (ISRv) and NFVIS as the hosting layer, the platform offers a complete solution for a simplified deployment. It also accelerates some functions in hardware, such as inter-VM traffic flows, IP Security (IPsec) crypto, and RAID for storage. Cisco SD-WAN capability can be enabled on both ENCS 5100 and 5400 appliances. Cisco SD-WAN offers an entirely new way to manage and operate a WAN infrastructure, offering better user experience, greater agility, and advanced

threat protection for branch office. Cisco ENCS is supported by Cisco DNA Center for management, orchestration, and automation.

Network Path Isolation

Path isolation refers to the creation of independent logical traffic paths over a shared physical network infrastructure. This involves the creation of VPNs with various mechanisms and mapping between various VPN technologies, Layer 2 segments, and transport circuits to provide end-to-end isolated connectivity between various groups of users.

The main goal when segmenting a network is to preserve and, often, improve the scalability, resiliency, and security services available in a non-segmented network. Any technology used to achieve virtualization must also provide the mechanisms necessary to preserve resiliency and scalability and to improve security.

A traditional hierarchical IP network is a combination of Layer 3 (routed) and Layer 2 (switched) domains. Both types of domains must be virtualized, and the virtual domains must be mapped to each other to keep traffic segmented. This can be achieved by combining the virtualization of the network devices (also referred to as *device virtualization*) with the virtualization of their interconnections (known as *data path virtualization*).

Layer 2 and Layer 3 Virtualization

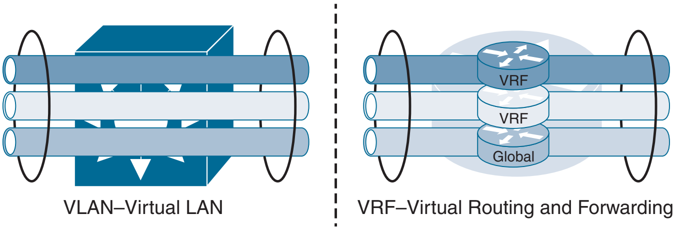
Virtualization in the Layer 2 domain is not a new concept: VLANs have been used for years. Within the campus, network virtualization began with the concept of VLANs. VLANs provide the most basic means of isolating network traffic at Layer 2 in a broadcast domain and require a Layer 3 device to route between those domains.

The goal of a robust network design is to minimize the extent of the broadcast domain and exposure to spanning tree loops. It is required to translate the Layer 2 VLAN to a Layer 3 virtual network or VPN. This Layer 3 VPN must be capable of supporting its own unique control plane completely, with its own addressing structure and routing tables for data forwarding that are completely isolated from any other Layer 3 VPN on that device and in the network. The technology that enables this type of functionality is known as virtual routing and forwarding (VRF).

A VRF instance is defined on a networking device that serves as the boundary between the Layer 2 (client-side) VLANs and the Layer 3 network. Each VRF instance consists of an IP routing table, a forwarding table, and the interface(s) assigned to it. Common routing protocols such as OSPF, EIGRP, BGP, and RIPv2 can be used to advertise and learn routes to populate the routing table that is unique to each virtual network. VRF instances can be compared to virtual routers co-resident on a single Layer 3 switch or router. Figure 7-10 illustrates Layer 2 virtualization (VLAN) and Layer 3 virtualization (VRF).

Data path virtualization is the virtualization of the interconnection between devices, which can be a single-hop or multi-hop interconnection. The type of data path virtualization varies depending on how far the VRF instances are from each other. If the virtualized devices are directly connected to each other (single hop), link or circuit virtualization is necessary. If the virtualized devices are connected through multiple hops over an IP network, a tunneling mechanism is necessary.

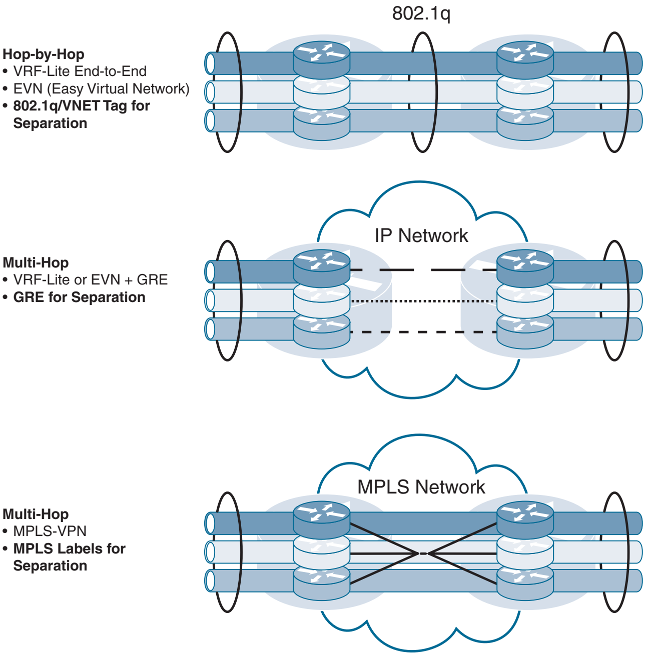
Figure 7-10 Layer 2 Virtualization and Layer 3 Virtualization


VRF
VRF
Global
VLAN–Virtual LAN
VRF–Virtual Routing and Forwarding

VRF-Lite combines the use of VRF instances with either 802.1Q trunking for hop-by-hop path isolation or Generic Routing Encapsulation (GRE)/Multipoint GRE (mGRE) for multi-hop path isolation. VRF-Lite with 802.1Q trunking would be found in a campus network where the IP routing is completely under the control of the organization deploying it and would typically make use of OSPF or EIGRP as the IGP for routing.

VRF-Lite can be used along with GRE or Multipoint GRE (mGRE) when it becomes necessary to extend a virtual network across a Layer 3 infrastructure or domain where virtualization is either not required or, as in the case of a service provider WAN, is beyond the control of the organization. In addition to GRE and mGRE, MPLS can also be used to extend VRFs across an IP infrastructure as well. These different options are illustrated in Figure 7-11.

Figure 7-11 Data Path Virtualization


802.1q
Hop-by-Hop
• VRF-Lite End-to-End
• EVN (Easy Virtual Network)
• 802.1q/VNET Tag for Separation
IP Network
Multi-Hop
• VRF-Lite or EVN + GRE
• GRE for Separation
MPLS Network
Multi-Hop
• MPLS-VPN
• MPLS Labels for Separation

In addition, within each networking device there are two planes to virtualize:

- **Control plane:** The control plane consists of all the protocols, databases, and tables necessary to make forwarding decisions and maintain a functional network topology free of loops or unintended black holes. This plane can be said to draw a clear picture of the topology for the network device. A virtualized device must have a unique picture of each virtual network it handles; thus, there is a requirement to virtualize the control plane components.

- **Forwarding plane:** The forwarding plane consists of all the processes and tables used to forward traffic. The forwarding plane builds forwarding tables based on the information provided by the control plane. Each virtual network has a unique forwarding table that needs to be virtualized.

The control and forwarding planes can be virtualized at different levels, which map directly to different layers of the OSI model. For instance, a device can be VLAN-aware and therefore can be virtualized at Layer 2, yet have a single routing table, which means it is not virtualized at Layer 3. The various levels of virtualization may or may not be useful, depending on the technical requirements of a deployment. In some cases, such as with a wiring closet, Layer 2 virtualization is enough. In other cases, virtualization of other layers may be necessary; for example, providing virtual firewall services requires Layer 2, 3, and 4 virtualization, plus the ability to define independent services on each virtual firewall, which perhaps is Layer 7 virtualization.

Virtual Routing and Forwarding

A virtual routing and forwarding (VRF) instance consists of an IP routing table, a derived forwarding table, a set of interfaces that use the forwarding table, and a set of rules and routing protocols that determine what goes into the forwarding table. The use of VRF technology allows a customer to virtualize a network device from a Layer 3 standpoint, creating different "virtual routers" in the same physical device.

The use of Cisco VRF-Lite technology has the following advantages:

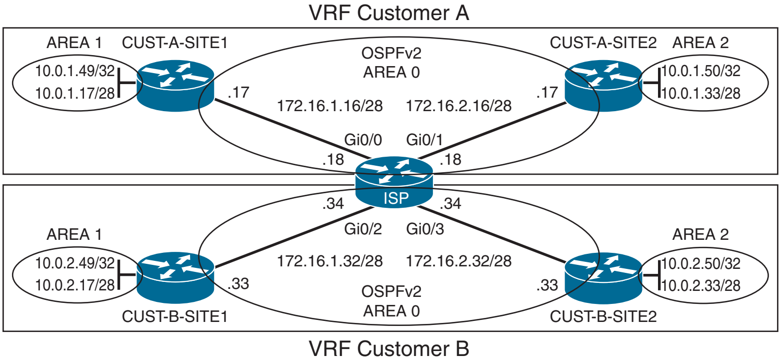
- **Allows for true routing and forwarding separation:** Dedicated data and control planes are defined to handle traffic belonging to groups with various requirements or policies. This represents an extra level of segregation and security because no communication between devices belonging to different VRF instances is allowed unless explicitly configured.
- **Simplifies the management and troubleshooting of the traffic belonging to the specific VRF because separate forwarding tables are used to switch that traffic:** These data structures are different from the one associated to the global routing table. This also guarantees that configuring the overlay network does not cause issues (such as routing loops) in the global table.
- **Enables support for alternate default routes:** The advantage of using a separate control and data plane is that it allows for the definition of a separate default route for each virtual network (VRF). This can be useful, for example, when providing guest access in a deployment when there is a requirement to use the default route in the global routing table just to create a black hole for unknown addresses to aid in detecting certain types of worm and network scanning attacks.

A VRF instance achieves the virtualization of the networking devices at Layer 3. VRF instances must be mapped to the appropriate VLANs at the edge of the network to provide continuous virtualization across the Layer 2 and Layer 3 portions of the network. The mapping of VLANs to VRF instances is as simple as placing the corresponding VLAN interface at the distribution switch into the appropriate VRF instance. The same type of mapping mechanism applies to Layer 2 virtual circuits (ATM, Frame Relay) and IP tunnels that a router handles as logical interfaces.

Configuring and Verifying VRF-Lite

The ISP router in Figure 7-12 is configured with VRF instances for Customer A and Customer B to achieve path isolation. The two customers use OSPFv2 for routing between their respective sites.

Figure 7-12 VRF-Lite Configuration Example


VRF Customer A
AREA 1
CUST-A-SITE1
10.0.1.49/32
10.0.1.17/28
.17
OSPFv2
AREA 0
172.16.1.16/28
172.16.2.16/28
CUST-A-SITE2
AREA 2
10.0.1.50/32
10.0.1.33/28
.17
Gi0/0
Gi0/1
.18
.18
ISP
.34
.34
Gi0/2
Gi0/3
AREA 1
10.0.2.49/32
10.0.2.17/28
172.16.1.32/28
172.16.2.32/28
AREA 2
10.0.2.50/32
10.0.2.33/28
.33
.33
OSPFv2
AREA 0
CUST-B-SITE1
CUST-B-SITE2
VRF Customer B

The use of separate routing tables solves part of the problems related to preventing one customer's packets from leaking into another customer's network because of overlapping prefixes, while allowing all sites for the same customer to communicate. A VRF instance exists inside a single router. Typically, a router needs at least one VRF instance for each customer attached to that router. In the example shown in Figure 7-12, the ISP router is connected to the two customers and must be configured with both VRF instances.

Each VRF instance has three main components:

- An IP routing table (RIB)
- A CEF FIB that is populated based on that VRF instance's RIB
- A separate instance or process of the routing protocol used to exchange routes with the CE devices that need to be supported by the VRF instance

Example 7-1 shows the commands required to deploy VRF-Lite for the scenario shown in Figure 7-12. This example assumes that the customer routers are already configured for OSPFv2.

Example 7-1 Configuring VRF-Lite

```
ISP(config)# ip vrf CUST-A
ISP(config-vrf)# exit
ISP(config)# ip vrf CUST-B
ISP(config-vrf)# exit
ISP(config)# interface GigabitEthernet 0/0
ISP(config-if)# ip vrf forwarding CUST-A
% Interface Ethernet0/0 IPv4 disabled and address(es) removed due to enabling VRF
  CUST-A
ISP(config-if)# ip address 172.16.1.18 255.255.255.240
ISP(config-if)# interface GigabitEthernet 0/1
ISP(config-if)# ip vrf forwarding CUST-A
% Interface Ethernet0/1 IPv4 disabled and address(es) removed due to enabling VRF
  CUST-A
ISP(config-if)# ip address 172.16.2.18 255.255.255.240
ISP(config-if)# interface GigabitEthernet 0/2
ISP(config-if)# ip vrf forwarding CUST-B
% Interface Ethernet0/2 IPv4 disabled and address(es) removed due to enabling VRF
  CUST-B
ISP(config-if)# ip address 172.16.1.34 255.255.255.240
ISP(config-if)# interface GigabitEthernet 0/3
ISP(config-if)# ip vrf forwarding CUST-B
% Interface Ethernet0/3 IPv4 disabled and address(es) removed due to enabling VRF
  CUST-B
ISP(config-if)# ip address 172.16.2.34 255.255.255.240
ISP(config-if)# exit

ISP(config)# router ospf 100 vrf CUST-A
ISP(config-router)# router-id 0.0.0.100
ISP(config-router)# network 172.16.1.16 0.0.0.15 area 0
ISP(config-router)# network 172.16.2.16 0.0.0.15 area 0
ISP(config-router)# router ospf 200 vrf CUST-B
ISP(config-router)# router-id 0.0.0.200
ISP(config-router)# network 172.16.1.32 0.0.0.15 area 0
ISP(config-router)# network 172.16.2.32 0.0.0.15 area 0
```

The first step is to configure the ISP router with a VRF instance for Customer A and a VRF instance for Customer B by using the **ip vrf** *vrf-name* command. Once the VRF instances are created, interfaces need to be assigned to each VRF instance by using the **ip vrf forwarding** *vrf-name* command. Notice that when an interface is assigned to a VRF instance, the router produces a warning message indicating that any IP address previously configured on the interface has been removed. This is because the interface has been moved from the global routing table to a specific VRF instance. The last step is to create two instances of OSPF, one for Customer A and one for Customer B, by using the **router ospf** *process-id* **vrf** *vrf-name* command. Each instance uses a unique OSPF process ID and a unique OSPF router ID.

To verify VRF-Lite, use the **show ip vrf** and **show ip vrf interfaces** commands, as shown in Example 7-2.

Example 7-2 Verifying VRF-Lite

```
ISP# show ip vrf
  Name                             Default RD           Interfaces
  CUST-A                           <not set>            Gi0/0
                                                        Gi0/1
  CUST-B                           <not set>            Gi0/2
                                                        Gi0/3

ISP# show ip vrf interfaces
Interface            IP-Address      VRF                             Protocol
Gi0/0                172.16.1.18     CUST-A                          up
Gi0/1                172.16.2.18     CUST-A                          up
Gi0/2                172.16.1.34     CUST-B                          up
Gi0/3                172.16.2.34     CUST-B                          up
```

Because the interfaces are now part of a VRF instance, you need to use the **show ip route vrf** *vrf-name* command to view specific VRF instance routing tables. The **show ip route** command will show you only the global routing table. The same applies to the **show ip protocols** command. You need to use the **show ip protocols vrf** *vrf-name* command to view VRF-specific routing information.

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	27
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	13

Day 6

Cisco DNA Center

ENCOR 350-401 Exam Topics

Network Assurance

- Describe Cisco DNA Center workflows to apply network configuration, monitoring, and managementu

Key Topics

Today we review concepts related to Cisco DNA Center, with a particular focus on its network automation and assurance features. Digitalization is disrupting businesses and organizations on a scale more like a tsunami than a rising tide. IT and business leaders are indicating that in order to compete in the face of such disruption, their organizations need to harness the power of digital technologies or risk being left behind. This market transition affects everyone, whether they are creating new experiences for consumers or their workforce, building operational technology for their business, or creating the next generation of services for their end users.

Cisco Digital Network Architecture (DNA) is an intent-based networking (IBN) solution that gives Cisco customers both a road map to digitalization and a path to recognize immediate benefits of network automation, assurance, and security across the WAN and the campus and branch networks.

Cisco DNA is a controller-based architecture. With Cisco DNA Center at its core, Cisco DNA provides the policy, automation, and analytics required to adapt to change, simplify and scale operations, and protect against degradation and threats. It provides 360-degree open platform extensibility to vastly increase the network's value by opening the way for the new and innovative applications, processes, and solutions that are required today and those that will be needed in the future.

Need for Digital Transformation

So, what exactly is digital transformation, and how does it affect a network?

Consider a simple taxi ride. Say that you are sitting in a hotel room, and tomorrow, you need to go to the airport so you can go home. You call down to the front desk and arrange for a pickup at a set time. The front desk calls the taxi dispatcher and arranges for the pickup. The next morning, the taxi company computer sends out the dispatch. A driver picks up the dispatch and hopefully arrives at your hotel at the proper time. You take the ride to the airport and face the hassle of paying and getting your receipt. If all the elements of this workflow function correctly, you are glad. If any element fails, you might miss your flight, and you have no way to register your complaints.

Enter companies such as Uber, Lyft, and Grab. They moved the dispatch process from the front desk and dispatcher to an app on your phone. You have no phone calls to make and no pickups to schedule. You choose your ride and set your location. You see a photo of your driver and details about the vehicle, and you can track their arrival on the map. You can even rate the driver and pay from the app. How does this example relate to your network? Say that you are the IT manager for the taxi company and want to upgrade to more digital services. Your current network is static, and your workflows are network-centric. To move to a more digital model, you must become user focused, offer more services, and be more adaptive. Your workflows must become more automated. You have a whole new set of priorities and requirements. The look and feel of the network are changing.

According to the Cisco Visual Networking Index (VNI), mobile traffic is now exceeding wired traffic. You need to support a user who is roaming not just from room to room but all over the world. In addition, the IoT is changing the face of the end user. The number of IoT devices is expected to triple in the near future, so more and more, end users are actually devices. Finally, annual global cloud IP traffic is expected to reach the zettabyte threshold, hitting 14.1 ZB. These changes bring a whole new set of service requirements. Because of the changing nature of the users and network requirements, you need a deeper understanding of network operations. You can no longer be concerned only about end-to-end connectivity and throughput. You have to think about new analytic requirements. Also, your network is going virtual, and the cloud is now more than just a storage point.

Digitalization is changing the way businesses, partners, employees, and consumers interact at an unprecedented pace. Products and services can be customized, ordered, and delivered with the click of a button using web-based applications. Business data can be acquired, analyzed, and exchanged in near real time. Geographic boundaries between businesses and consumers are diminishing. And the network is at the center of communication to and between the applications driving the digital economy.

How does an IBN help you in business terms? It allows you to continuously align your network to your organization's needs. It delivers contextual network-based insights, automates processes, and protects against threats. For example, it learns the performance needs of all your applications and services and continuously adapts end-to-end to achieve the required service quality. In addition, it automatically applies access policies between users, applications, and devices—consistently and anywhere across the network. And it understands operational compliance requirements and continuously verifies that golden software images and patches are applied.

What makes all this possible is a fundamental transformation occurring in how networks are built and run:

- Closed and hardware-centric models are giving way to open, programmable, and software-centric platforms that integrate with applications, systems, and domains beyond the network. A network needs to have the flexibility to quickly change in alignment with rapidly changing business objectives.
- Manual, repetitive command-line-interface-driven management is being largely superseded by policy-based automation. Networks need to be easier to configure, operate, and maintain in

the face of increasing scale and complexity. Current operational models are not scalable or sustainable.

- Perimeter-based, reactive security has been supplanted by network-embedded, context-aware security that reaches from the cloud to the enterprise edge, where the user or identity is located. Networks need to identify and neutralize security threats before they cause harm. Multi-cloud, IoT, and mobile adoption open up new threat vectors that a network needs to constantly protect against.
- Simple network visibility is morphing into predictive and contextual analytics for users, devices, applications, things, and the network at all points in time. Networks need to provide full visibility in terms of how the network is operating and providing assurance that it is supporting the desired business initiatives and achieving compliance. It is important to identify any discrepancies and recommend fixes.

Cisco IBN reflects all these changes. With this architecture, business and IT can become far more nimble and respond to business conditions more quickly and intelligently.

Cisco Digital Network Architecture

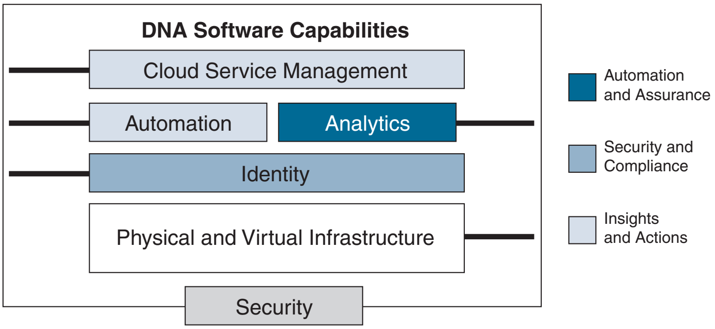
To compete in the digital era, customers must move to virtualized networks, enable controller-based automation, gather rich contextual analytics, deploy cloud-first strategies, and build on completely open environments. To support this move to virtualized networks, a layered architecture has been developed.

From its classic stack model look, you can see the roots of the developers in networking. At first glance, you might mistake it for a protocol stack such as TCP/IP. However, in this case, you have an architecture that is designed for functionality: virtualization, automation, analytics, cloud services, and network programmability.

Cisco Digital Network Architecture (DNA) is an open and extensible, software-driven architecture that is built on a set of design principles with the objective of providing the following, as shown in Figure 6-1:

- Insights and actions to drive faster business innovation. The architecture must not only manage the network but must also allow for the extraction of information related to network operations and end-user operations.
- Automation and assurance to reduce IT operational costs and complexity while meeting business and user expectations. This objective is accomplished through device and network virtualization, automation of configurations, and verification of quality of experience with assurance and feedback about possible automation of changes.
- Security and compliance to reduce risk as the organization continues to grow and open up to mobility, cloud, and other technologies. This objective is accomplished through a centralized and automated tool whose goal is to maintain compliance at the scale of the entire network more easily than a human.

Figure 6-1 Cisco DNA



The building blocks of Cisco DNA consist of the following:

- **Cloud service management:** This is possible with the Cisco DNA Center appliance, GUI, and related APIs. Developers use application programming interfaces (APIs) to expose features or functionalities in such a way that other developers can (re)use them in their applications. Within DNA, APIs are used so that applications within the building blocks can use functionality and features of other building blocks.
- **Automation:** The Cisco DNA Center appliance includes a built-in automation system that can be used to manage all network device configurations. Multiple mechanisms and protocols are used for configuration, such as SSH/CLI, SNMP, or NETCONF/YANG, and DNA Center uses one or more for each device type. All automation events, timestamps, and status are logged and can be reviewed in Cisco DNA Center or exported via APIs to IT service management (ITSM) tools.
- **Analytics:** The Cisco DNA Center appliance includes a built-in assurance system that manages analytics and telemetry. Key performance indicators (KPIs) are used to check whether the network, clients, and applications are running within set parameters. To ease troubleshooting, pervasive analytics (AI, machine learning, and other modern data analytics) are used to quickly ascertain specific behaviors and determine whether an incident is application, client, or network related.
- **Identity and virtualization:** Cisco Identity Services Engine (ISE) provides endpoint authentication and authorization information to Cisco DNA Center. Cisco ISE can interoperate with multiple sources of identity, such as Active Directory, LDAP, DNS records, and Amazon Web Services (AWS)/Microsoft Azure/Google Cloud Platform (GCP). The endpoint identity can then be used by the automation and analytics systems for policy configuration and assurance.
- **Infrastructure:** The infrastructure function comprises all network infrastructure–related devices that perform network functions, such as wired access, wireless access, router, WAN, and enforcing security policies. In essence, every network device—whether switch, router, access point, wireless LAN controller, firewall, or other security device, physical or virtual—is part of the infrastructure function.
- **Security:** Security is increasingly important and required for all aspects of the network. In Cisco DNA, security is tightly integrated within the design, with role-based access control, trusted certificates, and direct integration with Cisco ISE for authorization, identity, and

security policies. The correct security or application policy can be applied to an endpoint based on the identity of that endpoint (user and/or device).

Cisco Intent-Based Networking

Intent-based networking (IBN) offers a significant paradigm shift in how networks are planned, designed, and operated. In the past, tools were not available to declare intent and translate it into the device-level configurations required to realize a desired outcome. Instead, a network designer or operator had to manually derive individual network-element configurations to support the desired intent, such as, "I want these servers to be reachable from these branches; therefore, I need to configure specific VLAN, subnet, and security rules on each device in my network." Intent-based networking solutions enable conventional practices that require the alignment of manually derived individual network-element configurations to be replaced by controller-led and policy-based abstractions that easily enable operators to express intent (desired outcome) and subsequently validate that the network is doing what they asked of it.

Scale, agility, and security demands associated with digital transformation require that element-by-element network configuration be replaced by automated systemwide programming of network elements with consistent intent-based policies. Furthermore, the contextual analysis of data before, during, and after deployment enables continuous verification to help ensure that the network is delivering the desired outcomes and protection at any point in time. Continuous gathering of telemetry and other forms of data from a multitude of diverse sources provides a rich context of information to optimize a system and ensure that it is secure. Intent-based policy extends beyond the access control of clients or applications. It broadens to expressions of the desired user experience, application prioritization, service-chaining network functions that need to be applied to an application flow, or even operational service-level agreement (SLA) rules, such as "I want to deploy only golden images on my network devices."

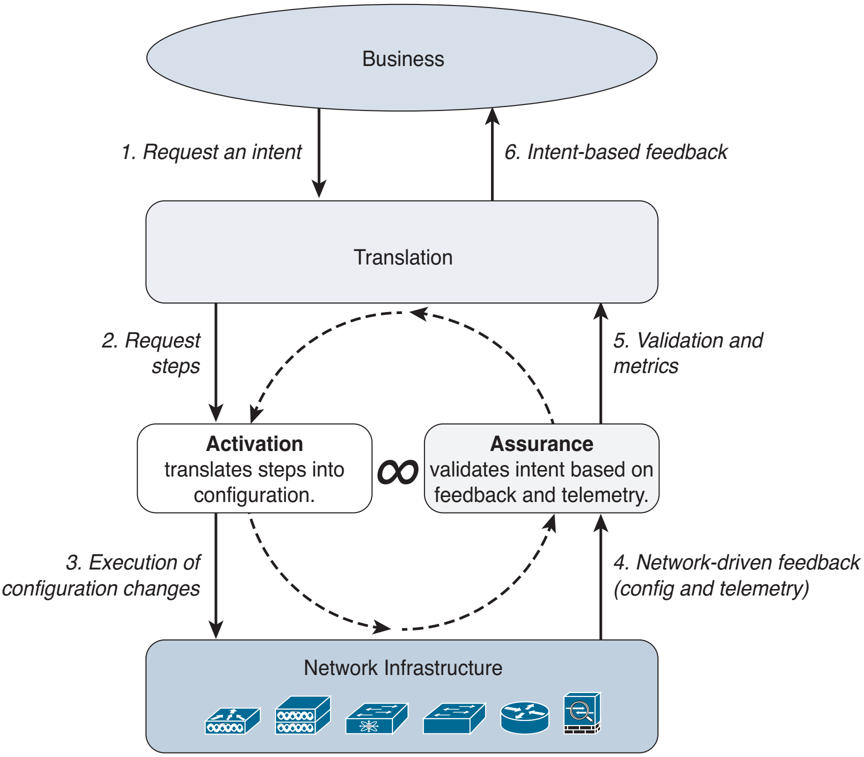
Today, implementing business requirements requires a lot of human interpretation and manual intervention to ensure that the IT systems are meeting these needs. Usually the process is lengthy, very resource intensive, and error prone. It does not meet the criteria for an agile digital business environment that has increasing numbers of systems, devices, applications, and services that need to be served. Intent-based networking captures the business intent, in business language, and translates this intent into automated IT policies that can be applied and constantly monitored across the network.

IBN Building Blocks

An intent-based network provides three principal functional buildings blocks, as shown in Figure 6-2: the capabilities to capture intent, functions to automate the deployment of the expressed intent throughout the network infrastructure, and the ability to ensure that the desired intent is being realized.

Although IBN is not based on Cisco Digital Network Architecture, its description and methodology are very similar to those of Cisco DNA. IBN describes how a network based on Cisco DNA can be configured and operated by the network operations team. Figure 6-2 illustrates the systematic approach IBN describes in providing intents to the network (by defining intents as repetitive pieces of configuration).

Figure 6-2 Intent-Based Approach to the Network


Business
1. Request an intent
6. Intent-based feedback
Translation
2. Request steps
5. Validation and metrics
Activation translates steps into configuration.
∞
Assurance validates intent based on feedback and telemetry.
3. Execution of configuration changes
4. Network-driven feedback (config and telemetry)
Network Infrastructure

IBN is based on these six steps, which occur in a continuous loop:

1. **Request intent:** Business or network operations request a specific intent.
2. **Request steps:** The intent is translated into a set of configuration changes to be executed.
3. **Execution of configuration changes:** Network configuration changes are executed via automation.
4. **Network-driven feedback:** The network infrastructure provides feedback on its operation.
5. **Validation and metrics:** The analytics component validates the received network-driven feedback with the requested intents to validate that the requested intents are operating as requested and designed.
6. **Intent-based feedback:** Business outcome–based values are used to report on the status of the requested intent and its operation.

The three main areas of IBN operation are translation, activation, and assurance:

- **Translation:** Capabilities that tell the network what to do to achieve the desired business outcome (intent), based on a consistent and verified policy the network can act upon.

  Example: Translate a business intent that the finance department needs highly secure and uninterrupted service levels for its month-end reporting application into network-relevant policies.

These policies may define the segmentation, security, and application service-levels for the network to implement.

- **Activation:** Deployment of the expressed policies throughout the network infrastructure by automating systemwide changes to all relevant network and security devices.

  Example: Apply priority service levels for all users and applications on the secure finance reporting segment across network and security devices in each network domain (data center, campus, WAN, and branch).

- **Assurance:** Continuous monitoring and verification to ensure that the desired intent has been applied and business outcome has been achieved. This can include remediation through recommended corrective actions and ongoing optimization through predictive analytics. Assurance includes (1) continuous verification of the IBN system behavior before, during, and after deployment; (2) deriving insights based on analytics (correlation of events and applying machine learning and artificial intelligence [ML/AI]) for validation, understanding, and prediction; (3) applying a closed-loop cycle to realize corrective action and optimization over time.

  Example: Use network telemetry to monitor and analyze finance application performance against desired outcomes, including remediation, optimization, and corrective actions, as appropriate.

Cisco DNA Center Features

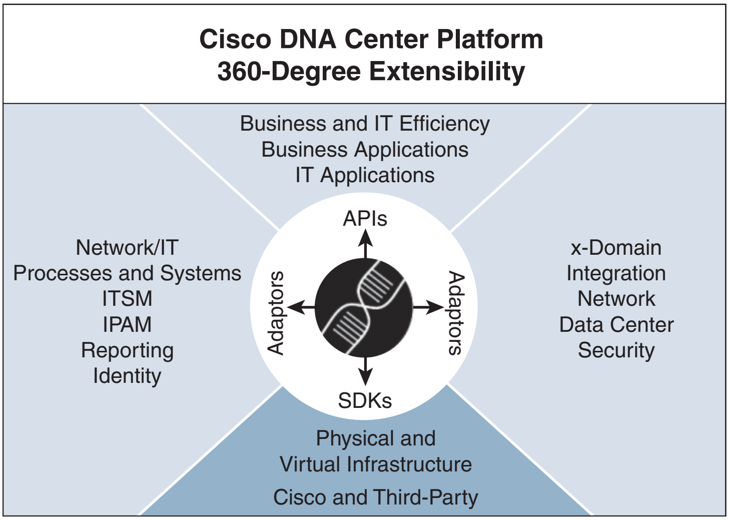
Cisco DNA Center is the foundational controller and analytics platform at the heart of Cisco's intent-based networking for large and midsize organizations. Cisco DNA Center provides a single dashboard for every fundamental management task to simplify network management. With this platform, IT can respond to changes and challenges faster and more intelligently.

Cisco DNA Center is a complete software-based network automation and assurance solution and is the dashboard for control and management of Cisco's intent-based networking system. Cisco DNA Center is also open and extensible. It provides broad support for external applications and systems to exchange data and intelligence, building on Cisco DNA Center's native functions. Using these open platform capabilities, applications can further ensure alignment of the organization's network with its business objectives.

Cisco DNA Center offers 360-degree extensibility through four distinct types of platform capabilities, as shown in Figure 6-3:

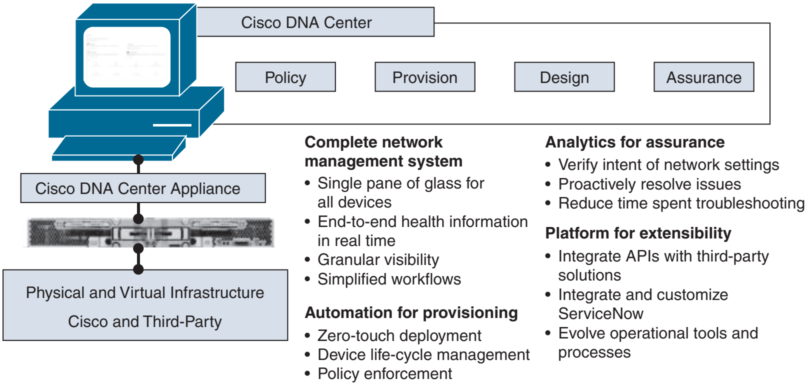
- Intent-based APIs enable business and IT applications to deliver intent to the network and to reap network analytics and insights for IT and business innovation.
- Process adapters, built on integration APIs, enable integration with other IT and network systems to streamline IT operations and processes.
- Domain adapters, built on integration APIs, allow integration with other infrastructure domains, such as data center, WAN, and security, to deliver a consistent intent-based infrastructure across the entire IT environment.
- Software development kits (SDKs) allow management to be extended to third-party vendors' network devices to offer support for diverse environments.

Figure 6-3 Cisco DNA Center Platform


Cisco DNA Center Platform
360-Degree Extensibility
Business and IT Efficiency
Business Applications
IT Applications
APIs
Network/IT
Processes and Systems
ITSM
IPAM
Reporting
Identity
Adaptors
Adaptors
x-Domain
Integration
Network
Data Center
Security
SDKs
Physical and
Virtual Infrastructure
Cisco and Third-Party

The following are some of the features and characteristics of Cisco DNA Center (see Figure 6-4):

Figure 6-4 Cisco DNA Center Features


Cisco DNA Center
Policy
Provision
Design
Assurance
Cisco DNA Center Appliance
Physical and Virtual Infrastructure
Cisco and Third-Party
Complete network management system
• Single pane of glass for all devices
• End-to-end health information in real time
• Granular visibility
• Simplified workflows
Automation for provisioning
• Zero-touch deployment
• Device life-cycle management
• Policy enforcement
Analytics for assurance
• Verify intent of network settings
• Proactively resolve issues
• Reduce time spent troubleshooting
Platform for extensibility
• Integrate APIs with third-party solutions
• Integrate and customize ServiceNow
• Evolve operational tools and processes

- **Design:** You can design a network using intuitive workflows, starting with locations where network devices will be deployed. Users of Cisco Prime Infrastructure and the Cisco Application Policy Infrastructure Controller Enterprise Module (APIC-EM) can simply import existing network designs and device images into Cisco DNA Center.
- **Policy:** You can define user and device profiles that facilitate highly secure access and network segmentation based on business needs. Application policies enable your business-critical applications to provide a consistent level of performance, regardless of network congestion.
- **Provision:** You can use policy-based automation to deliver services to the network based on business priority and to simplify device deployment. Zero-touch device provisioning and software image management features reduce device installation and upgrade time from hours to

minutes and bring new remote offices online with plug-and-play ease from an off-the-shelf Cisco device.

- **Assurance:** Cisco DNA Assurance enables every point on a network to become a sensor, sending continuous streaming telemetry on application performance and user connectivity in real time. This, coupled with automatic path trace visibility and guided remediation, means that network issues can be resolved in minutes—before they become problems. Integration with Cisco Stealthwatch security provides detection and mitigation of threats, even when the threats are hidden in encrypted traffic.
- **Platform:** An open and extensible platform allows third-party applications and processes to exchange data and intelligence with Cisco DNA Center. This improves IT operations by automating workflow processes based on network intelligence coming from Cisco DNA Center. Cisco DNA Center's open platform capabilities offer the next step in this evolutionary journey. These capabilities vastly increase the value to IT and the business by opening the way for new and innovative network-enabled applications, processes, and solutions that are required today, as well as those that will be identified in the future.

Cisco DNA Center is a software solution that resides on the Cisco DNA Center appliance. The solution receives data in the form of streaming telemetry from every device (switch, router, access point, and wireless access controller) on the network. This data provides Cisco DNA Center with the real-time information it needs for the many functions it performs.

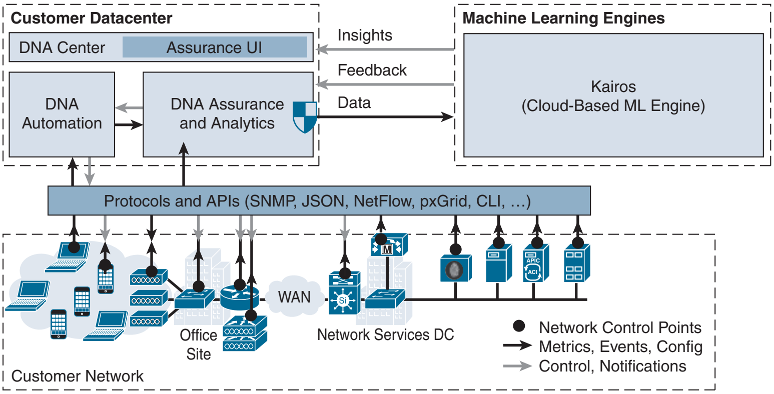
Cisco DNA Center Assurance

Cisco DNA Center Assurance provides a comprehensive solution to ensure better and consistent service levels to meet growing business demands. Assurance addresses not just reactive network monitoring and troubleshooting but the proactive and predictive aspects of running the network, ensuring client, application, and service performance.

Cisco DNA Center Assurance with artificial intelligence (AI) and machine learning eliminates manual troubleshooting operations and reduces the time spent resolving service issues. Cisco DNA Center Assurance turns network devices into sensors and then uses AI and machine learning to make sense of all the data. The clean and simple dashboard shows overall status and flags issues. Guided remediation then automates the process of issue resolution and performance enhancement to keep the user experience optimal, with less mundane troubleshooting work.

Cisco DNA Center Assurance resides within Cisco DNA Center and collects streaming telemetry from endpoints, devices, users, and applications across every part of a network, as illustrated in Figure 6-5. It uses Cisco AI network analytics to define what levels of performance are required for optimal user experience on the network and derive insights on optimization options. These insights are compared with the network policies in the Automation section of Cisco DNA Center to help ensure that the network operation is aligned with business intent. The results are delivered on uncluttered client and device health dashboards, where you can drill down for more detail on any section: clients, services, applications, wired devices, and wireless devices. Issues that require attention are flagged, and you can click on them for suggested issue remediation.

Figure 6-5 Cisco DNA Center Assurance Process


Customer Datacenter
DNA Center
Assurance UI
DNA Automation
DNA Assurance and Analytics
Insights
Feedback
Data
Machine Learning Engines
Kairos
(Cloud-Based ML Engine)
Protocols and APIs (SNMP, JSON, NetFlow, pxGrid, CLI, …)
WAN
Office Site
Network Services DC
M
APIC
ACI
Network Control Points
Metrics, Events, Config
Control, Notifications
Customer Network

Cisco DNA Center Assurance collects data from source types including NetFlow, traceroute, IP SLA, syslog, Cisco Web Security Appliance (WSA), RADIUS, Dynamic Host Configuration Protocol (DHCP), Active Directory and users, command-line interface and Secure Shell (CLI/SSH), NETCONFIG, pxGrid, DNS, AVC, Simple Network Management Protocol (SNMP), Cisco Connected Mobile Experiences (CMX), Apple iOS, and Samsung One UI.

Cisco DNA Center Automation Workflow

Cisco Digital Network Architecture offers centralized, intuitive management that makes it fast and easy to design, provision, and apply policies across the network environment. The Cisco DNA Center GUI provides end-to-end network visibility and uses network insights to optimize network performance and deliver the best user and application experience.

In this section, we review the features and applications supported for Cisco DNA Center automation. (All screen captures in this section are taken from the Cisco dCloud DNA Center Assurance 1.3.3 Instant Demo at dcloud.cisco.com.)

Network Discovery and Management

You can discover and manage your existing network by using the Discovery feature, which scans the devices in your network and sends a list of discovered devices to the Inventory function.

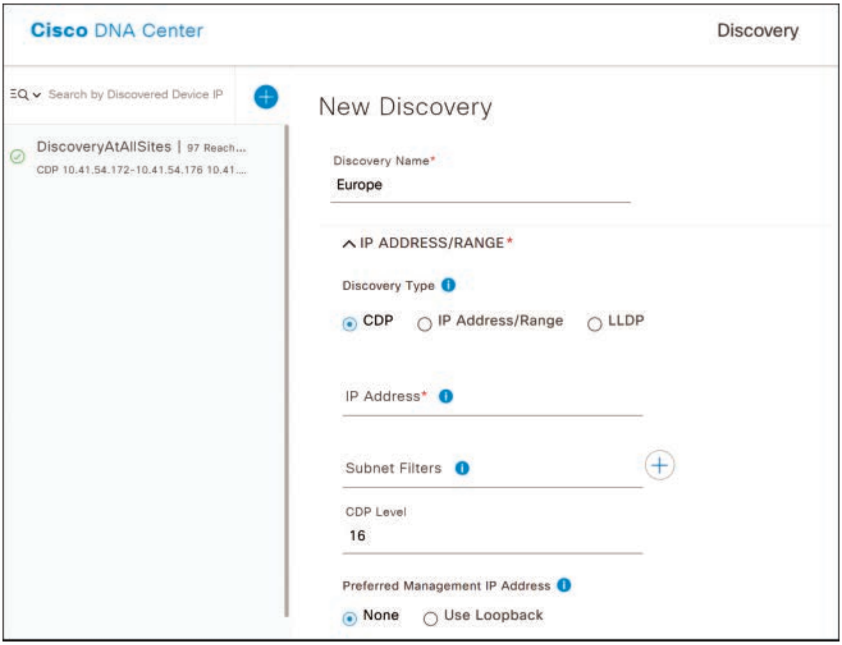
There are three ways to discover devices using Discovery:

- Use Cisco Discovery Protocol (CDP) and provide a seed IP address.
- Specify a range of IP addresses. (A maximum range of 4096 devices is supported.)
- Use Link Layer Discovery Protocol (LLDP) and provide a seed IP address.

Before you run Discovery, you must ensure that at least one SNMP credential is configured on your devices. Also, you need to configure SSH credentials on the devices you want Cisco DNA Center to discover and manage.

Figure 6-6 shows the Discovery process being initiated using CDP for the Europe zone.

Figure 6-6 Cisco DNA Center Discovery


Cisco DNA Center
Discovery
Search by Discovered Device IP

DiscoveryAtAllSites | 97 Reach...
CDP 10.41.54.172-10.41.54.176 10.41....

New Discovery

Discovery Name*
Europe

IP ADDRESS/RANGE*

Discovery Type
● CDP ○ IP Address/Range ○ LLDP

IP Address*

Subnet Filters

CDP Level
16

Preferred Management IP Address
● None ○ Use Loopback

Inventory Management

The Inventory function retrieves and saves details about devices—such as host IP addresses, MAC addresses, and network attachment points—in its database. Figure 6-7 shows a topology view of the Inventory function, and Figure 6-8 shows the list view drilled down to the San Jose 01 site (SJC01).

Figure 6-7 Cisco DNA Center Inventory: Topology View

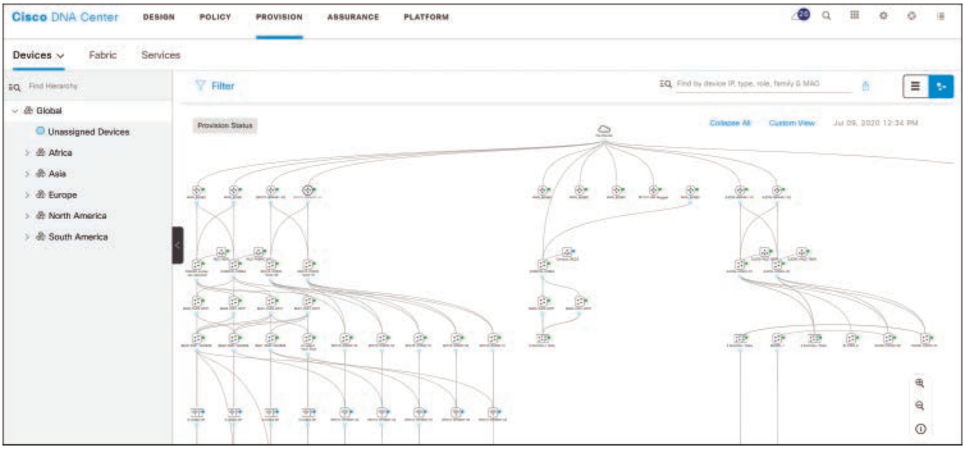
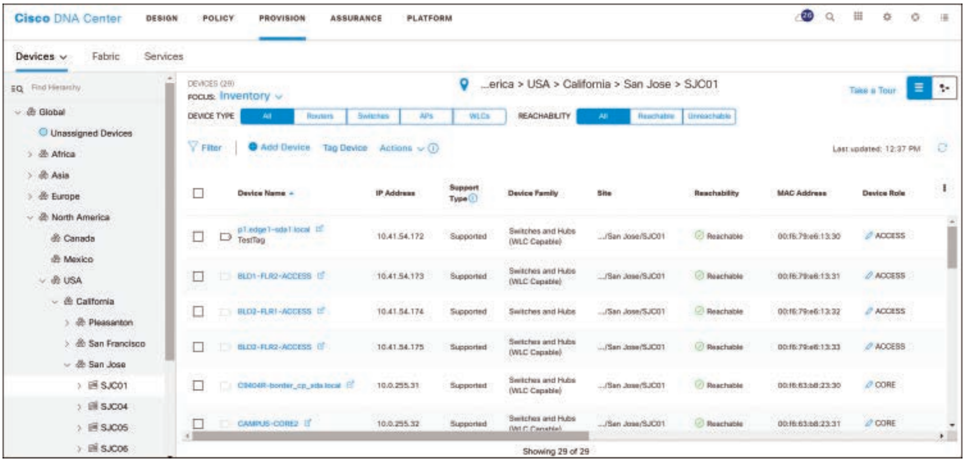

Cisco DNA Center
DESIGN
POLICY
PROVISION
ASSURANCE
PLATFORM
Devices
Fabric
Services
Find Hierarchy
Global
Unassigned Devices
Africa
Asia
Europe
North America
South America
Filter
Find by device IP, type, role, family & MAC
Provision Status
Collapse All
Custom View
Jul 09, 2020 12:34 PM

Figure 6-8 Cisco DNA Center Inventory: List View


Cisco DNA Center
DESIGN
POLICY
PROVISION
ASSURANCE
PLATFORM
Devices
Fabric
Services
Find Hierarchy
Global
Unassigned Devices
Africa
Asia
Europe
North America
Canada
Mexico
USA
California
Pleasanton
San Francisco
San Jose
SJC01
SJC04
SJC05
SJC06
DEVICES (29)
FOCUS: Inventory
...erica > USA > California > San Jose > SJC01
Take a Tour
DEVICE TYPE
All
Routers
Switches
APs
WLCs
REACHABILITY
All
Reachable
Unreachable
Filter
Add Device
Tag Device
Actions
Last updated: 12:37 PM

☐	Device Name	IP Address	Support Type	Device Family	Site	Reachability	MAC Address	Device Role
☐	p1.edge1-sda1.local TestTag	10.41.54.172	Supported	Switches and Hubs (WLC Capable)	.../San Jose/SJC01	Reachable	00:f6:79:e6:13:30	ACCESS
☐	BLD1-FLR2-ACCESS	10.41.54.173	Supported	Switches and Hubs (WLC Capable)	.../San Jose/SJC01	Reachable	00:f6:79:e6:13:31	ACCESS
☐	BLD2-FLR1-ACCESS	10.41.54.174	Supported	Switches and Hubs	.../San Jose/SJC01	Reachable	00:f6:79:e6:13:32	ACCESS
☐	BLD2-FLR2-ACCESS	10.41.54.175	Supported	Switches and Hubs (WLC Capable)	.../San Jose/SJC01	Reachable	00:f6:79:e6:13:33	ACCESS
☐	C9404R-border_cp_sda.local	10.0.255.31	Supported	Switches and Hubs (WLC Capable)	.../San Jose/SJC01	Reachable	00:f6:63:b8:23:30	CORE
☐	CAMPUS-CORE2	10.0.255.32	Supported	Switches and Hubs (WLC Capable)	.../San Jose/SJC01	Reachable	00:f6:63:b8:23:31	CORE

Showing 29 of 29

The Inventory feature can also work with the Device Controllability feature to configure the required network settings on devices, if these settings are not already present on the device. For more information about Device Controllability, see the *Cisco Digital Network Architecture Center Administrator Guide*.

Inventory uses the following protocols, as required:

- Link Layer Discovery Protocol (LLDP)
- IP Device Tracking (IPDT) or Switch-Integrated Security Features (SISF) (IPDT or SISF must be enabled on the device.)
- LLDP Media Endpoint Discovery (This protocol is used to discover IP phones and some servers.)
- Network Configuration Protocol (NETCONF)

After the initial discovery, Cisco DNA Center maintains the inventory by polling the devices at regular intervals. The default minimum interval is every 25 minutes. However, you can set this interval to up to 24 hours, as required for your network environment. Polling occurs for each device, link, host, and interface. On an average, polling 500 devices takes approximately 20 minutes.

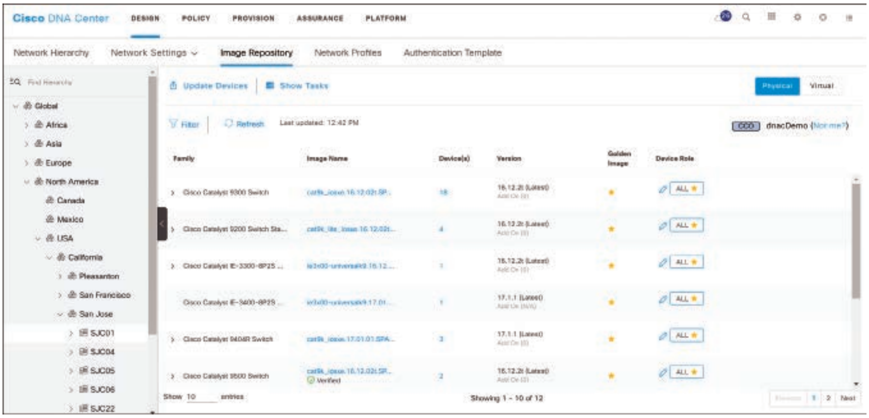
Software Image Management

Cisco DNA Center stores all the software images and software maintenance updates (SMUs) for the devices in your network. Image Repository provides the following functions:

- **Image storage:** Cisco DNA Center stores all the unique software images according to image type and version. You can view, import, and delete software images.
- **Provision:** You can push software images to the devices in your network.

Cisco DNA Center allows you to designate software images and SMUs as golden. A *golden* software image or SMU is a validated image that meets the compliance requirements for the particular device type. Designating a software image or SMU as golden saves you time by eliminating the need to make repetitive configuration changes and ensures consistency across your devices. Figure 6-9 shows the Image Repository window, where device families are associated with specific image names and versions.

Figure 6-9 Cisco DNA Center Software Image Repository


Cisco DNA Center
DESIGN
POLICY
PROVISION
ASSURANCE
PLATFORM
Network Hierarchy
Network Settings
Image Repository
Network Profiles
Authentication Template
Find Hierarchy
Global
Africa
Asia
Europe
North America
Canada
Mexico
USA
California
Pleasanton
San Francisco
San Jose
SJC01
SJC04
SJC05
SJC06
SJC22
Update Devices
Show Tasks
Physical
Virtual
Filter
Refresh
CCO
Previous
1
2
Next

Last updated: 12:42 PM

CCO dnacDemo (Not me?)

Family	Image Name	Device(s)	Version	Golden Image	Device Role
Cisco Catalyst 9300 Switch	cat9k_iosxe.16.12.02t.SP...	18	16.12.2t (Latest) Add On (0)	★	ALL ★
Cisco Catalyst 9200 Switch Sta...	cat9k_lite_iosxe.16.12.02t...	4	16.12.2t (Latest) Add On (0)	★	ALL ★
Cisco Catalyst IE-3300-8P2S ...	ie3x00-universalk9.16.12....	1	16.12.2t (Latest) Add On (0)	★	ALL ★
Cisco Catalyst IE-3400-8P2S ...	ie3x00-universalk9.17.01...	1	17.1.1 (Latest) Add On (N/A)	★	ALL ★
Cisco Catalyst 9404R Switch	cat9k_iosxe.17.01.01.SPA...	3	17.1.1 (Latest) Add On (0)	★	ALL ★
Cisco Catalyst 9500 Switch	cat9k_iosxe.16.12.02t.SP... Verified	2	16.12.2t (Latest) Add On (0)	★	ALL ★

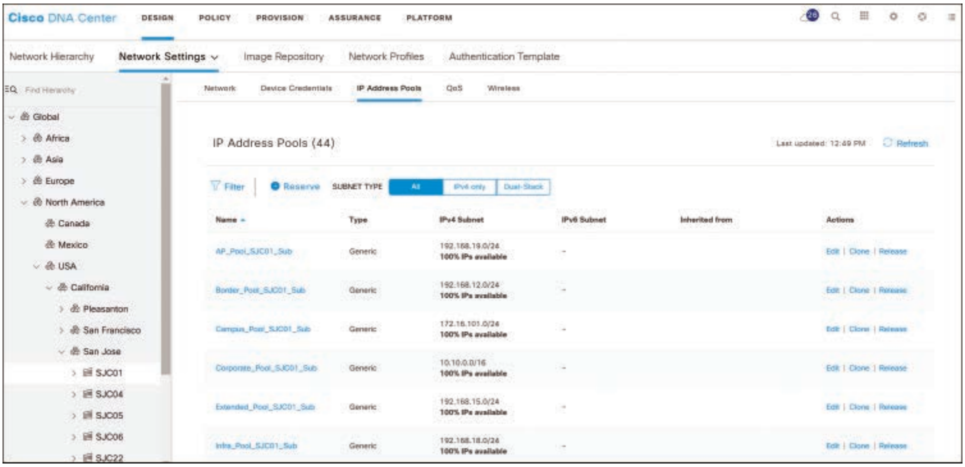
Show 10 entries

Showing 1 - 10 of 12

IP Address Pools

The Cisco DNA Center design tool enables you to either create new IP address pools or reserve pools that have already been learned from an integrated IP address management (IPAM) tool such as Infoblox. For customers that already use an IPAM solution like Infoblox, that tool can be integrated with Cisco DNA Center. If you are using other IPAM solutions, the REST APIs can be used for integration. You can also create a new IP address pool that can be managed centrally or at the site level. At the site level, you can further define IP address pools for LAN, management, and services, as shown in Figure 6-10.

Figure 6-10 Cisco DNA Center IP Address Pools


Cisco DNA Center
DESIGN
POLICY
PROVISION
ASSURANCE
PLATFORM
Network Hierarchy
Network Settings
Image Repository
Network Profiles
Authentication Template
Find Hierarchy
Global
Africa
Asia
Europe
North America
Canada
Mexico
USA
California
Pleasanton
San Francisco
San Jose
SJC01
SJC04
SJC05
SJC06
SJC22
Network
Device Credentials
IP Address Pools
QoS
Wireless
Filter
Reserve
SUBNET TYPE
All
IPv4 only
Dual-Stack
Refresh

IP Address Pools (44)

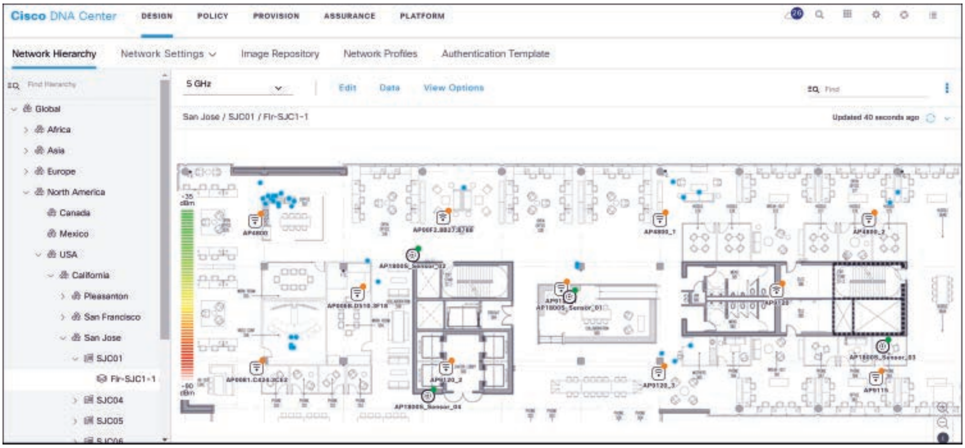
Last updated: 12:49 PM

Name	Type	IPv4 Subnet	IPv6 Subnet	Inherited from	Actions
AP_Pool_SJC01_Sub	Generic	192.168.19.0/24 100% IPs available	-		Edit | Clone | Release
Border_Pool_SJC01_Sub	Generic	192.168.12.0/24 100% IPs available	-		Edit | Clone | Release
Campus_Pool_SJC01_Sub	Generic	172.16.101.0/24 100% IPs available	-		Edit | Clone | Release
Corporate_Pool_SJC01_Sub	Generic	10.10.0.0/16 100% IPs available	-		Edit | Clone | Release
Extended_Pool_SJC01_Sub	Generic	192.168.15.0/24 100% IPs available	-		Edit | Clone | Release
Infra_Pool_SJC01_Sub	Generic	192.168.18.0/24 100% IPs available	-		Edit | Clone | Release

Network Hierarchy

The Cisco DNA Center design tool is built to support the scale of global customers while easily supporting customers with local or regional networks. With globally distributed sites, you face various challenges, such as networks being managed by regions, multiple network operations teams working on your network, and differences in network designs. With the hierarchical approach in network design, you can automate the rollout of regional changes while adhering to compliance standards throughout the network. For example, you can use the design tool and start at a global level by creating a country-level construct (for example, USA). Similarly, you can create global sites for any country where you have a network or set of networks that you want to provision. Then you can create an extra layer of hierarchy (such as a regional group of sites, like the western United States or a state) or simply add a building under the country-level site. The degree and level of hierarchy that you choose are up to you and your specific requirements. To add a building, you need to specify the site name and address; you can create very detailed location and mapping information, which is displayed in the global or regional view. You can also add building floor plan maps that can be imported to the design tool, as shown in Figure 6-11. This allows you to make changes such as defining floor properties, placement of APs, and so on.

Figure 6-11 Cisco DNA Center Network Hierarchy


Cisco DNA Center
DESIGN
POLICY
PROVISION
ASSURANCE
PLATFORM
Network Hierarchy
Network Settings
Image Repository
Network Profiles
Authentication Template
Find Hierarchy
Global
Africa
Asia
Europe
North America
Canada
Mexico
USA
California
Pleasanton
San Francisco
San Jose
SJC01
Flr-SJC1-1
SJC04
SJC05
5 GHz
Edit
Data
View Options
Find
San Jose / SJC01 / Flr-SJC1-1
Updated 40 seconds ago
-35 dBm
-90 dBm

Day 0 Network Provisioning

The Cisco Network Plug and Play application in Cisco DNA Center is used for device onboarding. Once they are powered on, devices securely connect to the Cisco DNA Center and need to be claimed using the Network Plug and Play application. A network administrator can claim the devices, and then Network Plug and Play downloads the certificates to add the sites for provisioning. This is important because not all Cisco devices can be provisioned using PnP.

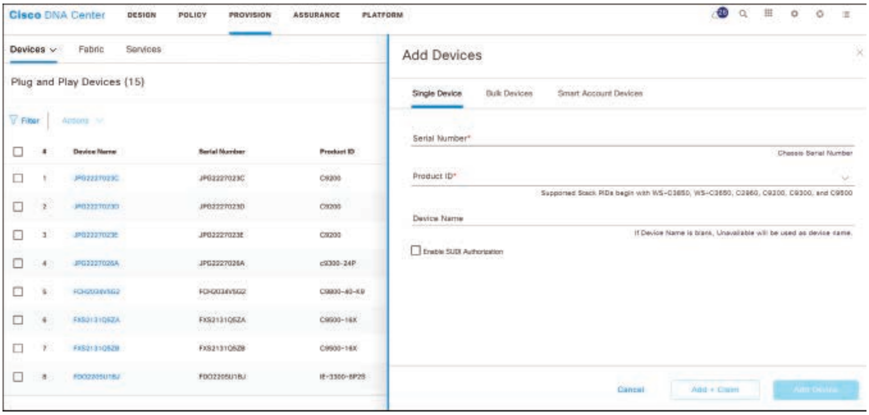
By using Network Plug and Play, you can do the following:

- Display the detailed onboarding status of network devices.
- Define provisioning workflows that include a series of actions, such as installing a software image and applying a device configuration.

- Plan devices before their installation by entering device information and associating a device with a workflow. When the device comes online, it contacts Cisco DNA Center, and the workflow provisions and onboards the device automatically.
- Provision unclaimed network devices, which are new devices that appear on the network without prior planning.
- Synchronize the device inventory from the Cisco Plug and Play Connect cloud portal in a Cisco Smart Account to the Network Plug and Play application so that all devices appear in Cisco DNA Center.

Figure 6-12 shows the Cisco DNA Center Plug and Play interface. Clicking Add Device allows a network administrator to enter serial numbers, product ID, and device name for quick onboarding and provisioning.

Figure 6-12 Cisco DNA Center Plug and Play Feature



Plug and Play Devices (15)

☐	#	Device Name	Serial Number	Product ID
☐	1	JPG2227023C	JPG2227023C	C9200
☐	2	JPG2227023D	JPG2227023D	C9200
☐	3	JPG2227023E	JPG2227023E	C9200
☐	4	JPG2227026A	JPG2227026A	c9300-24P
☐	5	FCH2034V5G2	FCH2034V5G2	C9800-40-K9
☐	6	FXS2131Q5ZA	FXS2131Q5ZA	C9500-16X
☐	7	FXS2131Q5ZB	FXS2131Q5ZB	C9500-16X
☐	8	FDO2205U1BJ	FDO2205U1BJ	IE-3300-8P2S

Serial Number*
Chassis Serial Number

Product ID*
Supported Stack PIDs begin with WS-C3850, WS-C3650, C2960, C9200, C9300, and C9500

Device Name
If Device Name is blank, Unavailable will be used as device name.

☐ Enable SUDI Authorization

Day *N* Network Automation

Network profiles are reusable network design templates that include site-agnostic parameters. Profiles offer standard device configuration based on "place in network" or "device role." For example, a WAN edge router can have a common configuration at multiple sites. The configuration may be based on the best practices and tested extensively, thus reducing the risk of manual errors. Once you have created the network profiles, they are assigned to the sites. The devices added to sites inherit the configuration defined in the profile.

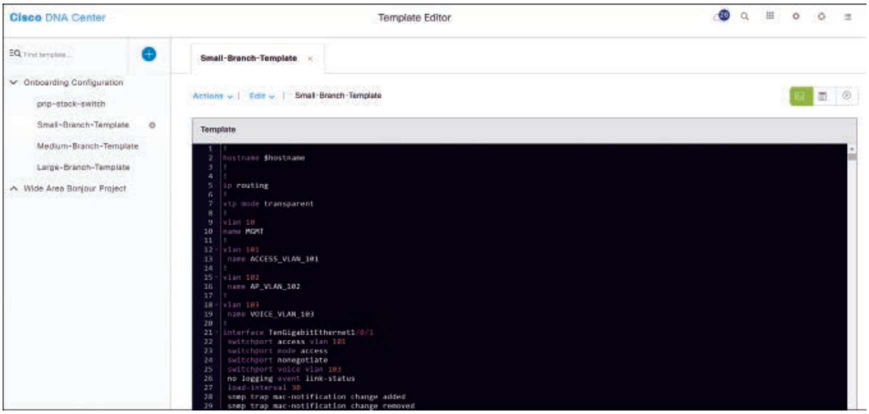
Cisco DNA Center provides an interactive editor to author CLI templates. Template Editor is a centralized CLI management tool to help design and provision templates in Cisco DNA Center. You can use a template to generate a device-deployable configuration by replacing the parameterized elements (variables) with actual values and evaluating the control logic statements. The Template Editor allows you to do the following with ease:

- Create, edit, and delete templates
- Add interactive commands
- Validate errors in the template

- Use version control templates for tracking purpose
- Simulate the templates

Figure 6-13 shows the Cisco DNA Center Template Editor with an example of a small branch template that could be used to automate the configuration of Layer 3 switches at those locations.

Figure 6-13 Cisco DNA Center Template Editor

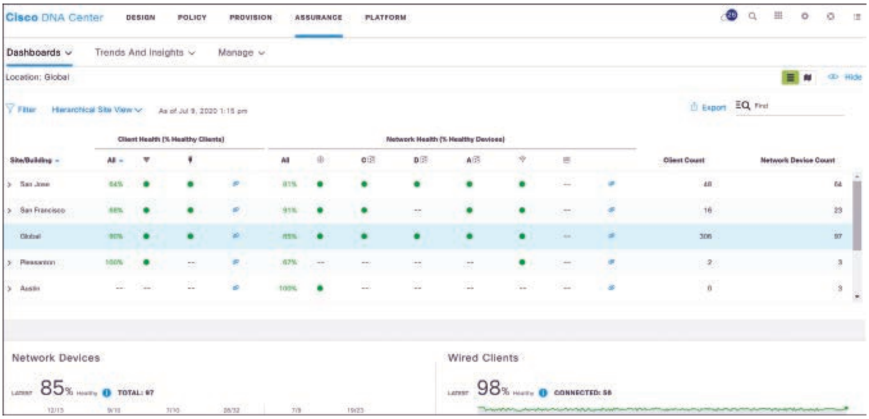

Cisco DNA Center
Template Editor
Find template...
Onboarding Configuration
pnp-stack-switch
Small-Branch-Template
Medium-Branch-Template
Large-Branch-Template
Wide Area Bonjour Project
Small-Branch-Template
Actions
Edit
Small-Branch-Template
Template

```
!
hostname $hostname
!
!
ip routing
!
vtp mode transparent
!
vlan 10
name MGMT
!
vlan 101
 name ACCESS_VLAN_101
!
vlan 102
 name AP_VLAN_102
!
vlan 103
 name VOICE_VLAN_103
!
interface TenGigabitEthernet1/0/1
 switchport access vlan 101
 switchport mode access
 switchport nonegotiate
 switchport voice vlan 103
 no logging event link-status
 load-interval 30
 snmp trap mac-notification change added
 snmp trap mac-notification change removed
```

Cisco DNA Assurance Workflow

Before you begin using the Assurance application, you must set up Cisco DNA Center to use Assurance. After installing Cisco DNA Center and discovering the network devices, you need to design the network hierarchy and configure the locations of devices, such as area, site, building, and floor. Once the devices are part of the inventory, you can assign them to the different regions/sites and start using Assurance. The Assurance dashboard can either display a list view or map view of devices, with their individual client health and network health percentages, as shown in Figures 6-14 and 6-15.

Figure 6-14 Cisco DNA Center Assurance Dashboard: List View

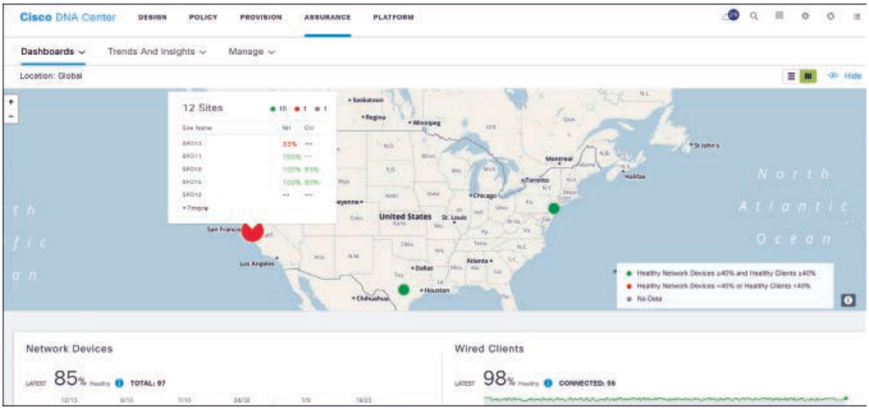

Cisco DNA Center
DESIGN
POLICY
PROVISION
ASSURANCE
PLATFORM
Dashboards
Trends And Insights
Manage
Location: Global
Hide
Filter
Hierarchical Site View
As of Jul 9, 2020 1:15 pm
Export
Find

	Client Health (% Healthy Clients)				Network Health (% Healthy Devices)									
Site/Building	All				All		C	D	A				Client Count	Network Device Count
San Jose	64%	●	●		81%	●	●	●	●	●	--		48	64
San Francisco	66%	●	●		91%	●	●	--	●	●	--		16	23
Global	90%	●	●		85%	●	●	●	●	●	--		306	97
Pleasanton	100%	●	--		67%	--	--	--	--	●	--		2	3
Austin	--	--	--		100%	●	--	--	--	--	--		0	3

Network Devices
Latest 85% Healthy
TOTAL: 97
Wired Clients
Latest 98% Healthy
CONNECTED: 58

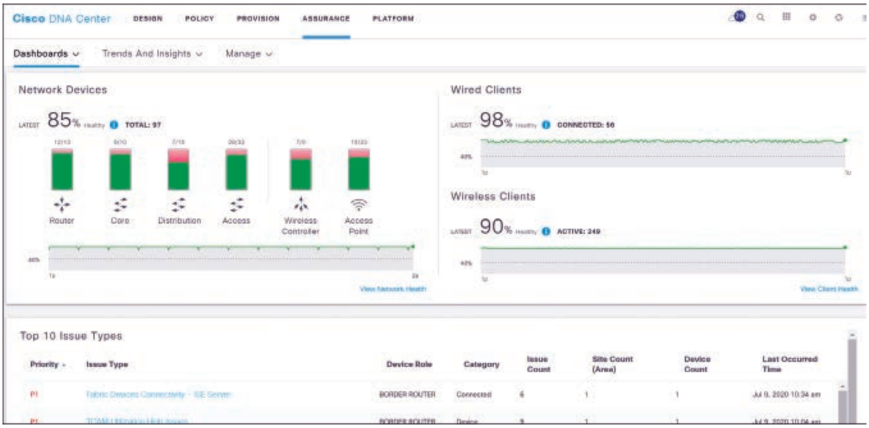
@cisco_in_persian_channel

Figure 6-15 Cisco DNA Center Assurance Dashboard: Map View


Cisco DNA Center
DESIGN
POLICY
PROVISION
ASSURANCE
PLATFORM
Dashboards
Trends And Insights
Manage
Location: Global
Hide
12 Sites
San Francisco
Los Angeles
United States
North Atlantic Ocean
Healthy Network Devices ≥40% and Healthy Clients ≥40%
Healthy Network Devices <40% or Healthy Clients <40%
No Data
Network Devices
LATEST 85% Healthy
TOTAL: 97
Wired Clients
LATEST 98% Healthy
CONNECTED: 56

The Assurance dashboard also displays general network information about network devices, wired and wireless clients, as well as the top 10 issues detected by Cisco DNA Center (see Figure 6-16).

Figure 6-16 Cisco DNA Center Assurance Dashboard: Network Health


Cisco DNA Center
DESIGN
POLICY
PROVISION
ASSURANCE
PLATFORM
Dashboards
Trends And Insights
Manage
Network Devices
LATEST 85% Healthy
TOTAL: 97
Router
Core
Distribution
Access
Wireless Controller
Access Point
View Network Health
Wired Clients
LATEST 98% Healthy
CONNECTED: 56
Wireless Clients
LATEST 90% Healthy
ACTIVE: 249
View Client Health

Top 10 Issue Types

Priority	Issue Type	Device Role	Category	Issue Count	Site Count (Area)	Device Count	Last Occurred Time
P1	Fabric Devices Connectivity - ISE Server	BORDER ROUTER	Connected	6	1	1	Jul 9, 2020 10:34 am

The device category health score (Access, Core, Distribution, Router, Wireless Controller, Access Point) is the percentage of healthy network devices (a health score from 8 to 10) in a target category, divided by the total number of network devices in that category. The score is calculated every 5 minutes. The health scores of devices are based on certain metrics, such as CPU utilization, memory utilization, and link errors. For example, an 85% health score = 82 network devices in a target category with health score from 8 to 10 across 97 network devices in total.

The Assurance dashboard also can show you the health history, which can help in proactively identifying issues. At the client level, you can see the application usage per business relevance category. The Assurance dashboard displays the health score for each application, with historical trends. You can gather detailed metrics such as throughput, packet loss, latency, and delay.

Network sensors use sensor-driven tests to test the health of wireless networks. A wireless network includes AP radios, WLAN configurations, and wireless network services. Assurance supports a dedicated sensor, which is dedicated hardware for performing sensor functions. The dedicated Cisco Aironet 1800s Active Sensor gets bootstrapped using PnP. After this sensor obtains Assurance server-reachability details, it directly communicates with the Assurance server.

The Assurance tool provides advanced search capabilities based on IP addresses, usernames, and so on. It provides a relevant 360-degree view of all details and issues experienced, which helps you in gathering information for troubleshooting.

Network Time Travel is another feature that allows you to go back in time to understand the network state when an issue occurred. It provides all information in history to show the critical events and identifies when an issue occurred.

Path Trace is a feature that allows you to follow the network path for any traffic flow from any source to the destination. You can gather detailed information for all devices and interfaces in the path. You can easily identify where access lists are blocking or affecting the data path.

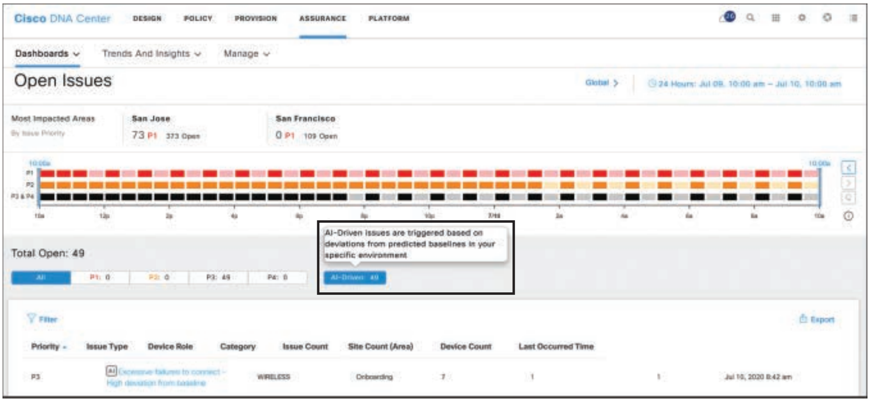
Application Experience (AppX) allows you to monitor the health of an application. Application health is measured using a score value that is calculated based on the application's qualitative metrics—packet loss and network latency. Based on the relevancy of an application and the NBAR standard, the application is classified as business relevant, business irrelevant, or default. Only application client/server statistics and application response time (ART) metrics can be monitored. Application Experience is based on the Cisco NetFlow records exported by the routers. To enable AppX, you must enable Cisco NetFlow collection on a device.

Cisco DNA Center Assurance: AI-Driven Data

Once a baseline has been established for normal network operation, it is possible for Cisco DNA Center Assurance to use AI-driven data to assist in resolving issues affecting network health.

After navigating to Assurance > Dashboards > Issues > Open, you can click the AI-Driven button, as shown in Figure 6-17. The 49 issues flagged in this example were triggered based on deviations from predicted baselines in the network. If you select one issue, you can drill down to see the problem details, the number of sites and clients that are impacted by the issue, and details related to the failure rates and baseline values at specific times.

Figure 6-17 Cisco DNA Center Assurance: AI-Driven Data



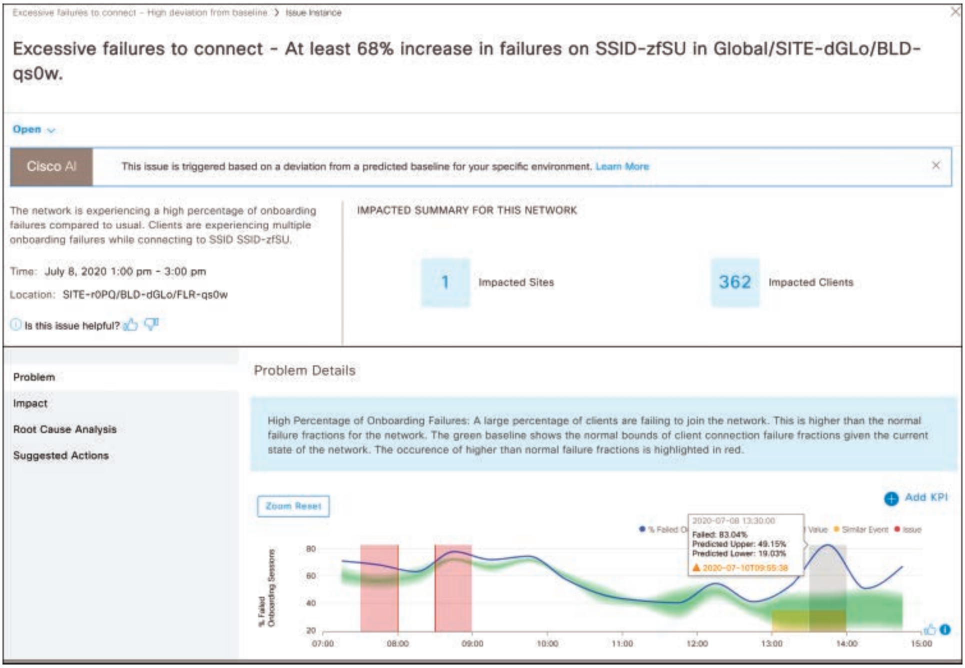
Cisco DNA Center
DESIGN
POLICY
PROVISION
ASSURANCE
PLATFORM
Dashboards
Trends And Insights
Manage
Open Issues
Global
24 Hours: Jul 09, 10:00 am – Jul 10, 10:00 am
Most Impacted Areas
By Issue Priority
San Jose
73 P1 373 Open
San Francisco
0 P1 109 Open
10:00a
P1
P2
P3 & P4
10a
12p
2p
4p
6p
8p
10p
7/10
2a
4a
6a
8a
10a
10:00a
Total Open: 49
All
P1: 0
P2: 0
P3: 49
P4: 0
AI-Driven 49
Filter
Export

AI-Driven issues are triggered based on deviations from predicted baselines in your specific environment

Priority	Issue Type	Device Role	Category	Issue Count	Site Count (Area)	Device Count	Last Occurred Time
P3	AI Excessive failures to connect – High deviation from baseline	WIRELESS	Onboarding	7	1	1	Jul 10, 2020 8:42 am

In the example in Figure 6-18, Cisco DNA Center Assurance is reporting that a large percentage of clients are failing to join the wireless network.

Figure 6-18 Cisco DNA Center Assurance: Baseline Deviation Reporting



Excessive failures to connect – High deviation from baseline › Issue Instance

Excessive failures to connect - At least 68% increase in failures on SSID-zfSU in Global/SITE-dGLo/BLD-qs0w.

Open

Cisco AI This issue is triggered based on a deviation from a predicted baseline for your specific environment. Learn More

The network is experiencing a high percentage of onboarding failures compared to usual. Clients are experiencing multiple onboarding failures while connecting to SSID SSID-zfSU.

Time: July 8, 2020 1:00 pm - 3:00 pm

Location: SITE-r0PQ/BLD-dGLo/FLR-qs0w

Is this issue helpful?

IMPACTED SUMMARY FOR THIS NETWORK

1 Impacted Sites

362 Impacted Clients

Problem Details

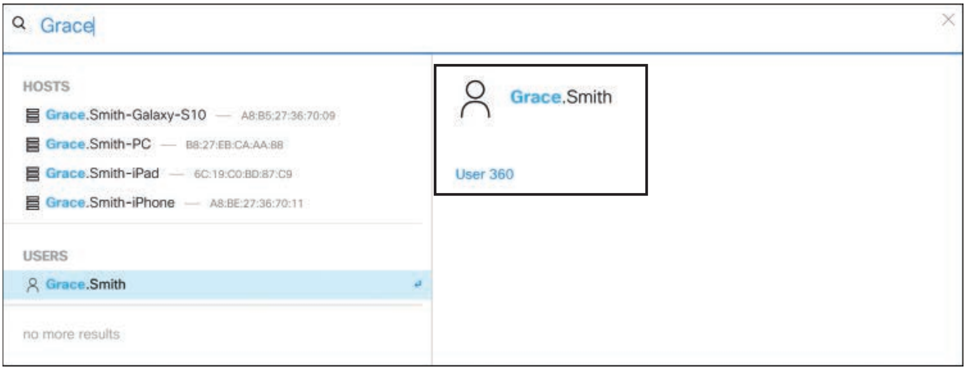
High Percentage of Onboarding Failures: A large percentage of clients are failing to join the network. This is higher than the normal failure fractions for the network. The green baseline shows the normal bounds of client connection failure fractions given the current state of the network. The occurence of higher than normal failure fractions is highlighted in red.

2020-07-08 13:30:00
Failed: 83.04%
Predicted Upper: 49.15%
Predicted Lower: 19.03%
2020-07-10T09:55:38

Cisco DNA Center Assurance: Client 360

When a user is reporting an issue, it is possible to quickly locate the user's devices by using the Cisco DNA Center Assurance Client 360 feature. In the example in Figure 6-19, the user Grace Smith is reporting wireless connectivity issues with one of her devices. A quick search in Cisco DNA Center shows that she currently has four devices visible on the network.

Figure 6-19 Cisco DNA Center Assurance: Client 360



Grace

HOSTS
Grace.Smith-Galaxy-S10 — A8:B5:27:36:70:09
Grace.Smith-PC — B8:27:EB:CA:AA:B8
Grace.Smith-iPad — 6C:19:C0:BD:87:C9
Grace.Smith-iPhone — A8:BE:27:36:70:11

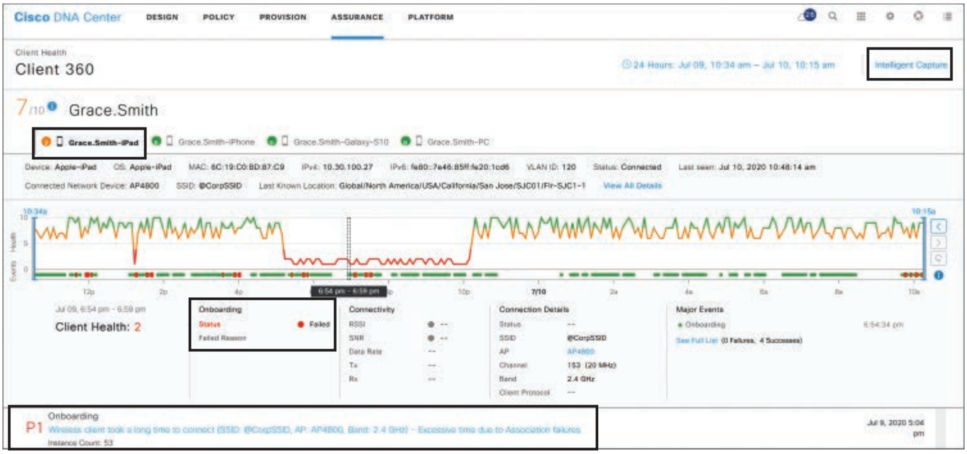
USERS
Grace.Smith

no more results

Grace.Smith

You can click User 360 to get a Client 360 view for all of Grace's devices, as shown in Figure 6-20. It appears that Grace's iPad was having onboarding issues from 6 p.m. to 10 p.m. The highlighted portion at the bottom of Figure 6-20 confirms the issue as well. By clicking the issue, you can further troubleshoot Grace's problem.

Figure 6-20 Cisco DNA Center Assurance: Client 360 Health Troubleshooting


Cisco DNA Center
DESIGN
POLICY
PROVISION
ASSURANCE
PLATFORM
Client Health
Client 360
24 Hours: Jul 09, 10:34 am – Jul 10, 10:15 am
Intelligent Capture
7/10
Grace.Smith
Grace.Smith-iPad
Grace.Smith-iPhone
Grace.Smith-Galaxy-S10
Grace.Smith-PC
View All Details
10:34a
10:15a
Health
Events
12p
2p
4p
6:54 pm - 6:59 pm
10p
7/10
2a
4a
6a
8a
10a
Jul 09, 6:54 pm - 6:59 pm
Onboarding
Connectivity
Connection Details
Major Events
P1
Jul 9, 2020 5:04 pm

Device: Apple-iPad OS: Apple-iPad MAC: 6C:19:C0:BD:87:C9 IPv4: 10.30.100.27 IPv6: fe80::7e46:85ff:fe20:1cd6 VLAN ID: 120 Status: Connected Last seen: Jul 10, 2020 10:48:14 am

Connected Network Device: AP4800 SSID: @CorpSSID Last Known Location: Global/North America/USA/California/San Jose/SJC01/Flr-SJC1-1

Client Health: 2

Onboarding

Status ● Failed

Failed Reason

Connectivity

RSSI ● --

SNR ● --

Data Rate --

Tx --

Rx --

Connection Details

Status --

SSID @CorpSSID

AP AP4800

Channel 153 (20 MHz)

Band 2.4 GHz

Client Protocol --

Major Events

● Onboarding 6:54:34 pm

See Full List (0 Failures, 4 Successes)

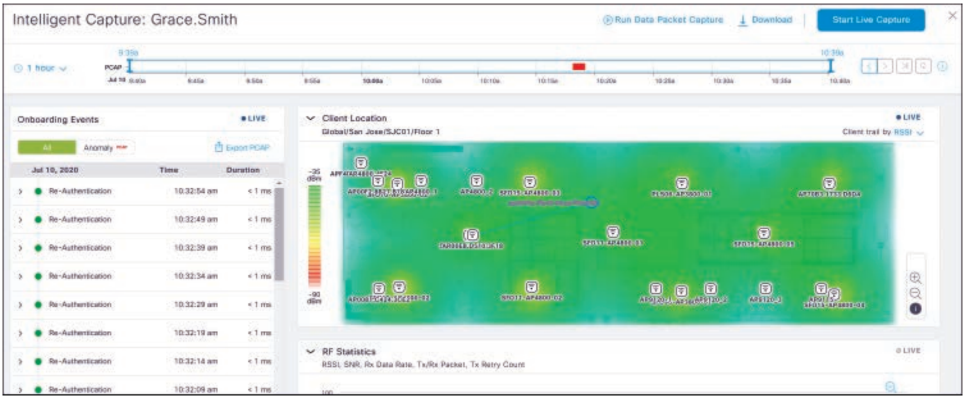
Onboarding

Wireless client took a long time to connect (SSID: @CorpSSID, AP: AP4800, Band: 2.4 GHz) – Excessive time due to Association failures

Instance Count: 53

Another option is to click Intelligent Capture at the top of the screen to examine the packet exchange between the AP and the client as the device moves through the building. As shown in Figure 6-21, the screen that appears allows you to select anomalous onboarding events to further troubleshoot the issue.

Figure 6-21 Cisco DNA Center Assurance: Client 360 Intelligent Capture

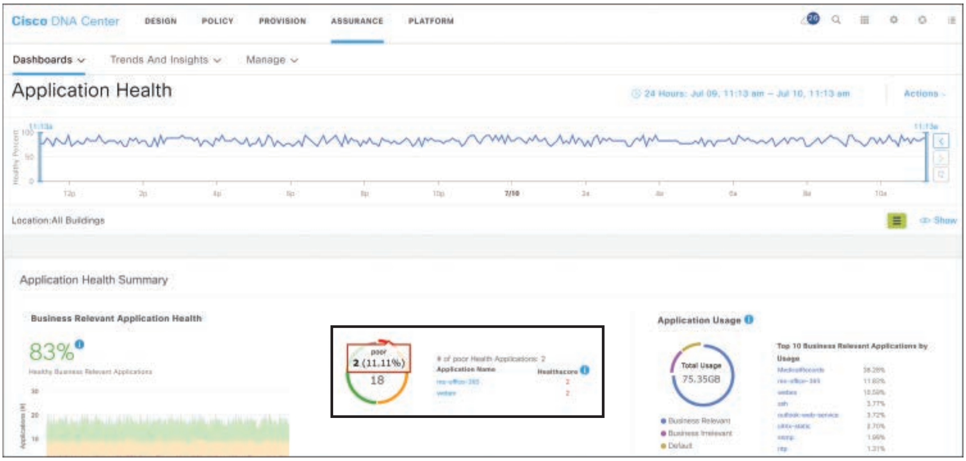

Intelligent Capture: Grace.Smith
Run Data Packet Capture
Download
Start Live Capture
1 hour
PCAP
9:39a
10:39a
Onboarding Events
● LIVE
All
Anomaly
Export PCAP
Client Location
● LIVE
Global/San Jose/SJC01/Floor 1
Client trail by RSSI
-35 dBm
-90 dBm
RF Statistics
LIVE
RSSI, SNR, Rx Data Rate, Tx/Rx Packet, Tx Retry Count
100

Jul 10, 2020	Time	Duration
● Re-Authentication	10:32:54 am	< 1 ms
● Re-Authentication	10:32:49 am	< 1 ms
● Re-Authentication	10:32:39 am	< 1 ms
● Re-Authentication	10:32:34 am	< 1 ms
● Re-Authentication	10:32:29 am	< 1 ms
● Re-Authentication	10:32:19 am	< 1 ms
● Re-Authentication	10:32:14 am	< 1 ms
● Re-Authentication	10:32:09 am	< 1 ms

Cisco DNA Center Assurance: Application Health

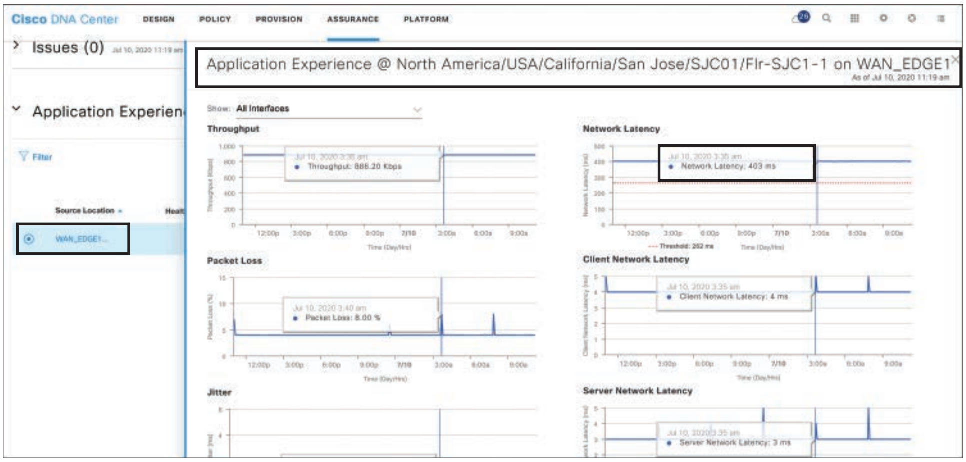
Another area that Cisco DNA Center Assurance can investigate is specific application health. If you navigate to Assurance > Dashboards > Health > Application Health, you see the Application Health dashboard, shown in Figure 6-22, which provides a quick overview of all business-critical applications.

Figure 6-22 Cisco DNA Center Assurance: Application Health Dashboard


Cisco DNA Center
DESIGN
POLICY
PROVISION
ASSURANCE
PLATFORM
Dashboards
Trends And Insights
Manage
Application Health
24 Hours: Jul 09, 11:13 am – Jul 10, 11:13 am
Actions
Location: All Buildings
Show
Application Health Summary
Business Relevant Application Health
83%
Healthy Business Relevant Applications
poor
2 (11.11%)
18
# of poor Health Applications: 2
Application Name
Healthscore
ms-office-365
2
webex
2
Application Usage
Total Usage
75.35GB
Business Relevant
Business Irrelevant
Default
Top 10 Business Relevant Applications by Usage

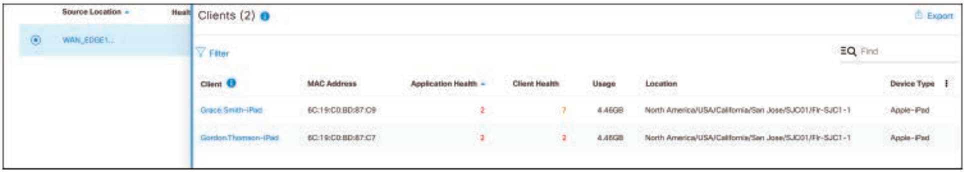
In this example, Cisco DNA Center Assurance is highlighting two applications that are in poor health: Microsoft Office 365 and Cisco Webex. By clicking the ms-office-365 link, you can see where in the network the poor health is being reported. In this case, the location is the WAN_EDGE1 router in San Jose, as shown in Figure 6-23.

Figure 6-23 Cisco DNA Center Assurance: Application Experience Reporting


Cisco DNA Center
DESIGN
POLICY
PROVISION
ASSURANCE
PLATFORM
Issues (0)
Application Experien
Filter
Source Location
Healt
WAN_EDGE1...
Application Experience @ North America/USA/California/San Jose/SJC01/Flr-SJC1-1 on WAN_EDGE1
As of Jul 10, 2020 11:19 am
Show: All Interfaces
Throughput
Jul 10, 2020 3:38 am
Throughput: 888.20 Kbps
Network Latency
Jul 10, 2020 3:35 am
Network Latency: 403 ms
Threshold: 262 ms
Time (Day/Hrs)
Packet Loss
Jul 10, 2020 3:40 am
Packet Loss: 8.00 %
Client Network Latency
Jul 10, 2020 3:35 am
Client Network Latency: 4 ms
Jitter
Server Network Latency
Jul 10, 2020 3:35 am
Server Network Latency: 3 ms

In this example, Cisco DNA Center is reporting high network latency and packet loss issues that are affecting two users, Grace Smith and Gordon Thomson, as shown in Figure 6-24.

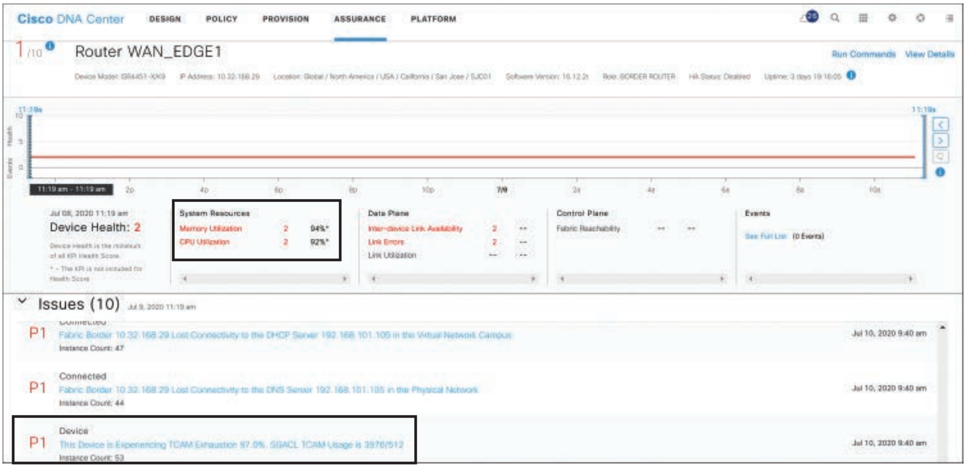
Figure 6-24 Cisco DNA Center Assurance: Users Affected by Poor Application Health



Client	MAC Address	Application Health	Client Health	Usage	Location	Device Type
Grace Smith-iPad	6C:19:C0:BD:87:C9	2	7	4.46GB	North America/USA/California/San Jose/SJC01/Flr-SJC1-1	Apple-iPad
Gordon Thomson-iPad	6C:19:C0:BD:87:C7	2	2	4.46GB	North America/USA/California/San Jose/SJC01/Flr-SJC1-1	Apple-iPad

Clicking the WAN_EDGE1 link takes you to the Cisco DNA Center Assurance Device 360 dashboard, which allows you to troubleshoot the issues affecting the device. Figure 6-25 shows that the WAN_EDGE1 router's memory and CPU utilization are close to 100%, and the device is also reporting 97% utilization of its TCAM. By clicking each issue affecting the router, you can get suggestions about possibly improving the device's health.

Figure 6-25 Cisco DNA Center Assurance: Device 360



1 /10 Router WAN_EDGE1

Device Model: ISR4451-X/K9 IP Address: 10.32.168.29 Location: Global / North America / USA / California / San Jose / SJC01 Software Version: 16.12.2t Role: BORDER ROUTER HA Status: Disabled Uptime: 3 days 19:16:05

Jul 08, 2020 11:19 am
Device Health: 2
Device Health is the minimum of all KPI Health Score
* - The KPI is not included for Health Score

System Resources		
Memory Utilization	2	94%*
CPU Utilization	2	92%*

Data Plane		
Inter-device Link Availability	2	--
Link Errors	2	--
Link Utilization	--	--

Control Plane		
Fabric Reachability	--	--

Events
See Full List (0 Events)

Issues (10) Jul 9, 2020 11:19 am

P1 Connected
Fabric Border 10.32.168.29 Lost Connectivity to the DHCP Server 192.168.101.105 in the Virtual Network Campus
Instance Count: 47
Jul 10, 2020 9:40 am

P1 Connected
Fabric Border 10.32.168.29 Lost Connectivity to the DNS Server 192.168.101.105 in the Physical Network
Instance Count: 44
Jul 10, 2020 9:40 am

P1 Device
This Device is Experiencing TCAM Exhaustion 97.0%. SGACL TCAM Usage is 3976/512
Instance Count: 53
Jul 10, 2020 9:40 am

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	23
Transforming Campus Networks to Intent-Based Networking	4, 5

Day 5

Network Programmability

ENCOR 350-401 Exam Topics

Network Assurance

- Configure and verify NETCONF and RESTCONF

Automation

- Interpret basic Python components and scripts
- Construct valid JSON encoded file
- Describe the high-level principles and benefits of a data modeling language, such as YANG

Key Topics

Today we review the basics of Python programming and its uses in networks today. As the enterprise network grows, traditional command-line management becomes more challenging and error prone. Implementing automation helps a network administrator become more efficient in managing network operations. Automation can be implemented in multiple ways; one way is to leverage network-centric programming. To better understand this, we review different data encoding formats, including Extensible Markup Language (XML) and JavaScript Object Notation (JSON), as well as the YANG data modeling language. We finish today by looking at management protocols such as Network Configuration Protocol (NETCONF) and Representational State Transfer Configuration Protocol (RESTCONF).

Python Concepts

In this section, we investigate why learning Python is important. We look at the two main versions of Python, how to execute code, and how to find help while working in the Python shell.

Python is one of the most popular and commonly used programming languages in the world. It is used by computer programmers and by people in various professions who are interested in automating and scripting tasks and making the workflows more efficient. Python is considered a relatively simple language that is easy to learn. Cisco has been putting Python on certain network devices and releasing many tools using Python, so it is a great choice for network engineers looking to add network programmability to their skill set.

Python's syntax is similar to the syntax of the English language, which makes Python relatively easy to learn for network engineers who have little or no programming background. Python has efficient, high-level data structures and a simple yet effective approach to object-oriented programming.

Python is an interpreted language, which means it executes instructions directly, without the need to compile the program first. It is also very flexible in its use, which makes it suited to writing various applications. You can use Python to write backend applications, and you can also use it to write web-enabled applications.

It is worth noting that Cisco now includes a Python execution engine on switch platforms, such as Nexus. Python scripts are therefore portable as they can run on a server, on a workstation, or locally on a switch.

Two versions of Python are currently in widespread use: Version 2.x and Version 3.x.

Python Version 2.x is currently supported by Cisco NX-OS, has extensive libraries, and is still supported by the Python community, although it is no longer in development. Python Version 2.x has a better support library than Version 3.x and is the default on Linux and Mac.

Python Version 3.x is currently under active development and is easier to learn than the earlier versions. While Python Version 3.x did fix some major issues that were known in Version 2.x, some features are not backward compatible with Version 2.x.

Because many libraries exist only for Version 2.x, this is the version that is still used widely by the network community.

Execute Python Code

There are two main ways to execute Python code:

- Using the Python dynamic interpreter
- Writing Python scripts

In this section we examine the various aspects of executing Python code using the Python shell and using the Python execution engine to write Python scripts.

Using the Dynamic Interpreter

The Python interactive interpreter, commonly known as the *Python shell*, is used to write and test code in real time, without having to write a full program or script. The Python shell is available natively on nearly all Linux distributions and Cisco switch platforms, such as Nexus.

To open the Python shell, enter the **python** command in either a Linux terminal or from the command line of a Cisco Nexus switch, as shown in Example 5-1 (which uses a Cisco Nexus 9300v switch). Because Python is built into the platform, no text editor or integrated development environment is necessary to start writing Python code. This is another reason Python has a lower barrier to entry as a programming language compared to other languages.

Example 5-1 Launching the Cisco NX-OS Version 9.3(3) Python Shell

```
n9kv# python
Python 2.7.11 (default, Feb 26 2018, 03:34:16)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()
n9kv#
```

To exit the Python shell, use **exit()**, **quit()**, or Ctrl+D.

Writing Python Scripts

Python scripts can be written in any text editor and saved locally with the .py file extension. When you run a script in this way, you can invoke the script simply by invoking the **python** statement and the filename.

Example 5-2 shows a Linux terminal prompt. First, the **touch** command is used to create a blank file called encor.py. The **gedit** command launches a text editor that allows the **print** command to be added to the file. After exiting the text editor, the **encor.py** program is executed using the **python** command, and the text is displayed in the terminal window.

Example 5-2 Writing Python Scripts

```
cisco@cisco:~$ touch encor.py
cisco@cisco:~$ gedit encor.py

print "Welcome to the world of Cisco Network Programmability!"

cisco@cisco:~$ python encor.py
Welcome to the world of Cisco Network Programmability!
cisco@cisco:~$
```

Note that you can access free online Python IDEs at https://www.programiz.com/python-programming/online-compiler/ and https://www.jdoodle.com/python3-programming-online/.

Python Helper Utilities and Functions

Several built-in Python helper utilities and functions can help when working at the command line and are shown in Example 5-3. This example assumes that the **python** command has already been entered. The example starts with the creation of a variable called **hostname** that is then assigned the string value **'n9kv'**. Then the **print** function confirms that the router's name is stored in the variable.

The **dir** command, when used without any arguments, returns a list of all the attributes in the current global scope. When you pass an object to **dir**, you see it return the list of attributes of that given object. When the **dir** command is used with the **hostname** variable, the shell displays all the attributes available for that variable.

The **type** command is used to see what the type of the argument is, such as string, integer, or dictionary. In this case, the **hostname** variable is type string.

The **help** command interprets the embedded documentation straight from the Python file you need help on. Here the **upper** function is explored as it relates to the **hostname** string variable. This function converts a string to all uppercase—so the **hostname.upper()** command converts the string **n9kv** to **N9KV**.

Example 5-3 Exploring the NX-OS Python Shell

```
>>> hostname = 'n9kv'
>>> print hostname
n9kv
>>>
>>> dir(hostname)
['__add__', '__class__', '__contains__', '__delattr__', '__doc__', '__eq__',
  '__format__', '__ge__', '__getattribute__', '__getitem__', '__getnewargs__',
  '__getslice__', '__gt__', '__hash__', '__init__', '__le__', '__len__', '__
  lt__', '__mod__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_
  ex__', '__repr__', '__rmod__', '__rmul__', '__setattr__', '__sizeof__',
  '__str__', '__subclasshook__', '_formatter_field_name_split', '_formatter_
  parser', 'capitalize', 'center', 'count', 'decode', 'encode', 'endswith',
  'expandtabs', 'find', 'format', 'index', 'isalnum', 'isalpha', 'isdigit',
  'islower', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip',
  'partition', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit',
  'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title',
  'translate', 'upper', 'zfill']
>>>
>>> type(hostname)
<type 'str'>
>>> help(hostname.upper)
Help on built-in function upper:

upper(...)
    S.upper() -> string

    Return a copy of the string S converted to uppercase.

>>> hostname.upper()
'N9KV'
>>>
```

Writing Idiomatic Python

Idiomatic Python programming is concerned with readability and uniformity. There are many things to consider when writing idiomatic Python, often referred to as *being Pythonic*.

Here are a few things to consider when writing Python to achieve readability and uniformity:

- **Single-line comments:** When creating a single-line comment, simply begin the line with the hash (#) character, and the comment automatically terminates at the end of the line.

- **Multi-line comments:** When multi-line comments (comments that span multiple lines) are needed, use a delimiter (a unique character that is not in the comment itself) to define the beginning and end of the comment.
- **Whitespace:** Whitespace characters such as spaces and tabs can be used for spacing.
- **Indentation (spaces vs. tabs):** Leading whitespace at the beginning of a line can help show the grouping of statements.
- **The Python Style Guide (PEP8):** PEP8 is a list of coding conventions for the Python code.

If you are contributing to open-source projects, you should thoroughly read the PEP8 style guide, which defines the recommended practices.

Common Python Data Types

Python makes use of many data types, as shown in Table 5-1.

Table 5-1 Common Python Data Types

Data Type	Definition
String	A sequence of character data
Number	An integer that allows for mathematical operation to be performed directly in code
List	A container used to store different types of data at the same time
Dictionary	A container that uses key/value pairs to match keys to their values
Boolean	A value or an operator used in code
File	An object used by Python

Each of the data types that Python utilizes provides unique aspects that make writing code easier. Depending on how the data is to be used, you can make use of various data types. To reference a sequence of data, you can put it into a string by surrounding the data with quotes or double quotes. For integers that need to be used in mathematical operations, you can use numbers. Values and operators that are used in code are referred to as *Booleans*. Python can use various objects that are referenced in files. To reference many objects at a time, you can create a list that contains a sequence of comma-separated objects within square brackets. Dictionaries allow you to create key/value pairs to match keys to their values.

String Data Types

A string is simply a sequence of characters that are surrounded by either quotes or double quotes. Without quotes, you are referencing an object. With quotes, it is a string. Strings are also immutable, meaning that after a string is defined, individual characters of the string cannot be modified. Here are some examples of strings:

```
ipaddr = '10.1.1.1'
mask = "255.255.255.0"
hostname = "nk9v"
password = ''
```

An empty string is simply a string that contains no characters. Note that any empty (no null) variable evaluates to **False** in a conditional statement and that a variable that is assigned with any value evaluates to **True** in a conditional statement.

Printing Strings

As you saw in Example 5-3, to print a string, you can use the **print** command. It is also possible to simply type the variable name in the interpreter. The main difference between these two methods is that the **print** command interprets escape characters, such as **\n**, which is a newline character, whereas the variable name typed at the command prompt displays all the options in text form. Example 5-4 shows a variable called **interface_config** being created. When the variable is called at the interpreter prompt, it prints the value as a string literal, whereas when the variable is printed using the **print** statement, the string is rendered, and the **/n** is taken into account.

Example 5-4 Printing Strings

```
>>> interface_config = 'interface Gi1/1\n no switchport'
>>>
>>> interface_config
'interface Gi1/1\n no switchport'
>>>
>>> print interface_config
interface Gi1/1
 no switchport
>>>
```

Concatenating Strings

You may often need to add one or more strings together. This process, called *concatenation*, is accomplished by using the **+** command. To combine two strings representing an IP address and a subnet mask into one new string, use the syntax shown in Example 5-5.

Example 5-5 Concatenating Strings

```
>>> ipmask = ipaddr + ' ' + mask
>>> print ipmask
10.1.1.1 255.255.255.0
>>>
```

Notice in Example 5-5 that when you add the two strings **ipaddr** and **mask**, a string containing a single space is added in between to represent a single whitespace in between the IP address and subnet mask.

There are several very commonly used built-in methods for working with strings. For example, you can use the **dir()** function on any object to see its available built-in methods and attributes, as shown in Example 5-3 with the **dir(hostname)** command.

String Built-in Methods

When working with strings, there are functions to modify or verify data within a string. These functions are called *methods*, and there are many methods available.

Python has several functions that allow a programmer to easily modify or manipulate strings. These functions are defined in the Python programming language and are ready for use. These ready-to-use functions are referred to as *built-in* methods. Example 5-6 illustrates several of the built-in methods that can be used with strings.

Example 5-6 Strings Methods

```
>>> macaddr = '00:11:22:33:44:55'
>>> macaddr.replace(':', '.')
'00.11.22.33.44.55'
>>>
>>> ipaddr = '10.1.1.1'
>>> ipaddr.startswith('10')
True
>>>
>>> ipaddr = '10.{}.1.1'
>>> ipaddr.format('200')
'10.200.1.1'
>>>
>>> ipaddr = '10.4.8.1'
>>> ipaddr.split('.')
['10', '4', '8', '1']
```

Example 5-6 first uses the **replace** method to change the **:** symbol to a period (**.**). Next, it uses the **startswith** method to verify that the IP address starts with the value **10**. Then it uses the **format** method and the **{}** symbols to insert the value **200** in the IP address. Finally, it uses the **split** method to create a list from the values separated by the **.** character.

Note that the **split** method returns a list, which is a data type covered later today. For now you can see that it is possible to use **split** to create a list of objects separated with a character; in this case the character used is a period (**.**) to simplify working with IP addresses. Each element in the list is numbered starting with 0. So, in Example 5-6, element number 2 would be the value **8**.

Numbers Data Types

In Python, you can perform mathematical operations directly in code by using integers and floating point numbers. Floating point numbers, also called *floats*, represent real numbers. A float is written with a decimal point dividing the integer and the fractional parts of the number, and an integer is written as just the number itself. You can use typical mathematical operators such as +, - , *, /, and // to represent the function being utilized. Example 5-7 illustrates some mathematical operations at the Python interpreter prompt.

Example 5-7 Mathematical Operations

```
>>> 5*8
40
>>>
>>> 1082780 + 435463
1518243
>>>
>>>
>>> 144 / 12
12
>>>
```

Boolean Data Types

In programming, a Boolean is a binary variable representing either true or false. Operators are **and**, **or**, and **not**. When using **and** in an expression, all values must be true for the expression to evaluate to True. It is nearly the opposite when using **or**; with **or**, only one value needs to be true to have the expression evaluate to True. The **not** operator produces the inverse of the Boolean result. Example 5-8 shows some results using Boolean values.

Example 5-8 Boolean Values

```
>>> True and False
False
>>>
>>> True or False
True
>>>
>>> not (True or False)
False
>>>
```

Describe Conditionals

In Python, conditional statements are used to perform different computations or actions, depending on whether a condition evaluates to True or False. Several types of operators are used to evaluate whether a condition is True or False:

- **Comparison operators:** ==, !=, >, <, >=, <=
- **Logical operators:** and, or, not
- **Membership operators:** in, not in
- **Identity operators:** is, is not

Example 5-9 shows some examples of conditional operators in use.

Example 5-9 Conditionals

```
>>> 9372 > 9396
False
>>>
>>> 'nexus' != 'catalyst'
True
>>>
>>> 'nexus' in 'nexus 9396'
True
>>>
>>> '9372' not in 'nexus 9396'
True
>>>
```

In Python, a conditional statement ends with a colon and must maintain consistent indentation to work properly. The three conditional statements are **if**, **elif**, and **else**. Example 5-10 shows an example of conditionals using the **if**, **elif**, and **else** statements.

Example 5-10 Conditionals Example

```
>>> switch = 'catalyst 3850'
>>>
>>> if 'catalyst' in switch:
...     switch_type = 'catalyst'
... elif 'nexus' in switch:
...     switch_type = 'nexus'
... else:
...     switch_type = 'unknown'
...
>>>
>>> print switch_type
catalyst
>>>
```

In Example 5-10, note the following:

- The **if** statement states a condition to be evaluated.
- **elif** is an optional condition that can be used numerous times to check for multiple expressions of True.
- **else** is the final statement and contains the code to execute when nothing is True.

When there is a match in an **if**/**elif**/**else** statement, the code block is exited.

Script Writing and Execution

Writing scripts is one of the basic programming skills of Python. Python is an interpreted programming language, which means you need only create the file, and then the file can be run. The file does not need to be compiled (translated) into a form that the machine understands; Python can interpret the file itself.

The following are important points to keep in mind when creating Python scripts:

- It is a good idea to include a shebang (used by the shell), as described in the next section.
- It is a good idea to define an entry point, as discussed shortly.
- Script code uses the same format as the Python interactive interpreter.
- Python files often end with a **.py** file extension.
- Python scripts are executed as **python** [*scriptname*]**.py**.

Shebang

It is recommended to use a shebang as the first line of a script. While doing so is not required, it does aid in enforcing which Python version is used by the shell environment when executing a given script.

A shebang, when used, must be the first line in the script. It is also the only line that can start with a # symbol that is not a comment. # symbols are used for inline comments anywhere else in a script.

For example, the shebang **#!/usr/bin/env python** tells the device executing the script to use the version of Python that is specified in the environment (**env**) path.

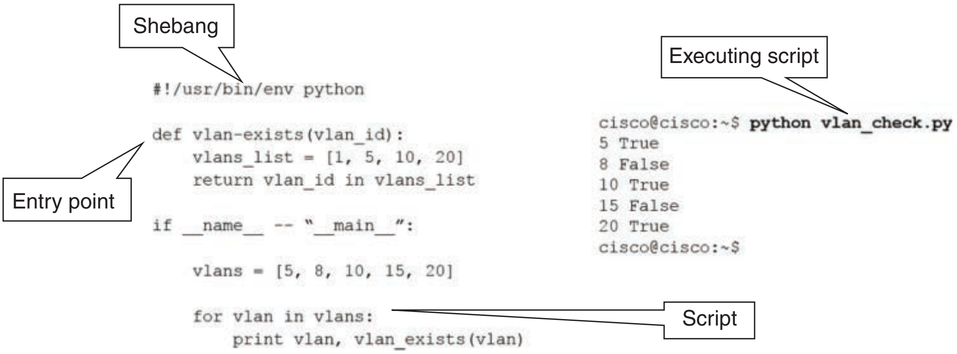
If you did not include the shebang, you can still check to see which version would be used by using the Linux command **which python**. You can also hard-code a version of Python to be used in the script—for example, Python Version 2.7 or Python Version 3.3. However, it is recommended to use the shebang **#! /usr/bin/env python** as it uses the version of Python that is configured in your Linux environment. This also allows you to easily change the version of Python being used without having to modify each individual script.

Script Entry Point

It is recommended to define an entry point for Python scripts. This means you can explicitly define where the code begins to get executed when the Python file is executed as a standalone program/script. Defining an entry point is optional for Python; if you do not define one, your code is executed from the top down when the Python file is run as a standalone file and when it's being used as a Python module.

Figure 5-1 shows the placement of the shebang and the code script's entry point. Note that the script is on the left side of the figure, and on the right side you can see the script being executed at the Linux terminal prompt.

Figure 5-1 Python Script Writing and Execution



```
#!/usr/bin/env python

def vlan-exists(vlan_id):
    vlans_list = [1, 5, 10, 20]
    return vlan_id in vlans_list

if __name__ -- "__main__":

    vlans = [5, 8, 10, 15, 20]

    for vlan in vlans:
        print vlan, vlan_exists(vlan)
```

```
cisco@cisco:~$ python vlan_check.py
5 True
8 False
10 True
15 False
20 True
cisco@cisco:~$
```

As you can see in Figure 5-1, everything else is the same as if you were working in the dynamic Python interpreter. The entry point and the shebang are the only differences, and both of them are optional.

Device Management and Network Programmability

Networks began to be managed 25+ years ago when they first started being deployed. In those days, networks were deployed manually and via the command line, just as they mostly are today. However, there were far fewer devices to deploy and manage, and there were also far fewer features than there are today, and so it was not necessarily obvious that Simple Network Management Protocol (SNMP) or CLI management would ultimately not be the right choice for device management.

In fact, those mechanisms served most environments just fine. SNMP still dominates the market for monitoring basic up/down status, and the CLI is still the primary means of management. However, neither of these options is optimal today.

SNMP has been around since the 1980s. It has historically been the de facto way to monitor networks. It worked great when networks were small, and polling a device every 15 to 30 minutes met operational requirements. However, SNMP often caused operational issues when polling devices too frequently. While it did work great for monitoring devices every few minutes, SNMP never caught on industry-wide for configuration management due to custom MIBs. A Management Information Base (MIB) is a type of data model that defines a collection of information organized in a hierarchical format that is used along with SNMP. SNMP uses a *get request* to retrieve a MIB variable from the SNMP agent (a network device), and it uses a *set request* to set variables that are stored in the MIB.

While SNMP has served the industry reasonably well from a device-monitoring perspective, it does have plenty of weaknesses. In particular, it does not support a network-friendly way to "program" the network using various languages such as Python, C, Java, and Ruby. True model-driven APIs simplify the generation and use of different languages.

As mentioned earlier, the CLI has also been available for network management but has some shortcomings. Access to the CLI typically happens via console, Telnet, or SSH. The CLI was meant for human-to-machine communication, not machine-to-machine communication. While the CLI will continue to play an integral role in troubleshooting and operations, it is error-prone and extremely inefficient for configuration management.

Based on the shortcomings of the current configuration management approaches, we can outline the requirements for next-generation configuration management:

- **Provide easier-to-use management interfaces:** We need clear and well-defined ways to interact with newer interfaces (APIs). We need to be able to leverage custom and open-source tools to easily consume the APIs.
- **Support client-side validation and error checking:** APIs should support the ability to offer client-side validation. When using a next-generation approach that is purely model driven, this becomes a great by-product. Rather than have a device do error checking, the management applications that leverage the device's API can automatically handle it.
- **Separate configuration and operational data:** There is a delineation between configuration state and operational state, and it needs to be reflected in the API. Any attribute, configuration parameter, or statistic should be accessible via the API.
- **Contain a built-in backup/restore capability:** Next-generation APIs should make it simpler to perform backups and restores, and they should also improve how changes are made, enabling them to be staged and applied as a transaction.
- **Be both human and machine friendly:** APIs need to be easier for humans to consume. APIs that support human-readable data formats such as JSON and XML simplify adoption. Data-encoded documents that are derived from data models further improve the machine readability and improve the pace at which changes can be deployed on a device and, in turn, reflected and understood by humans.

As next-generation programmatic interfaces are being built, a few key attributes must be considered:

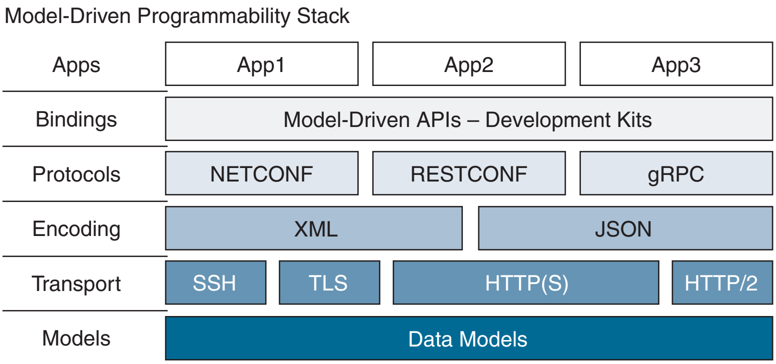
- These interfaces must support different types of transport, including HTTP, SSH, and TLS.
- They must be flexible and support different types of encoding formats, such as XML and JSON.
- We need efficient and easy-to-use tools that help consume the new APIs (for example, programming libraries).
- There must be extensible and open APIs, such as REST, RESTCONF, NETCONF, and gRPC.
- They must be model driven. Being model driven improves transport, API, encoding, and data formats.

Figure 5-2 illustrates the model-driven programmability stack. The core components of the complete device API include the following:

- **Data models:** Data models are the foundations of APIs. Data models define the syntax and semantics, including constraints of working with an API.

- **Transport:** Model-driven APIs support one or more transport methods, including SSH, TLS, and HTTP(s).
- **Encoding:** Model-driven APIs support a choice of encoding, including XML and JSON, and also custom encodings, such as Google protocol buffers.
- **Protocols:** Model-driven APIs support multiple options for protocols, with the three core protocols being NETCONF, RESTCONF, and gRPC. REST is not explicitly listed here because when REST is used in a modeled device, it becomes RESTCONF. (We examine REST later today.)

Figure 5-2 Model-Driven Programmability Stack


Model-Driven Programmability Stack
Apps
App1
App2
App3
Bindings
Model-Driven APIs – Development Kits
Protocols
NETCONF
RESTCONF
gRPC
Encoding
XML
JSON
Transport
SSH
TLS
HTTP(S)
HTTP/2
Models
Data Models

Data Encoding Formats

Two main data encoding formats are commonly used in APIs: JSON and XML. Each of them provides a structured way of using data formatting to send data between two computer systems. This is in stark contrast to using SSH and issuing CLI commands, with data sent as raw text, such as strings, over the wire.

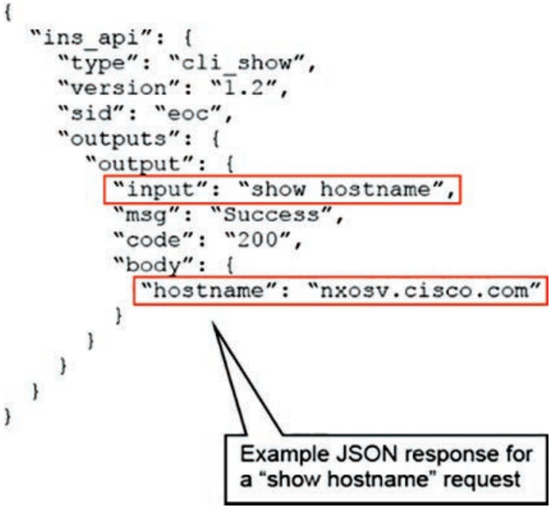
JSON

JavaScript Object Notation (JSON) is a data exchange format and open standard. It is easy for humans to read and write, and it is easy for machines to parse and generate. It is based on a subset of JavaScript and is a text format that is completely language independent. JSON sends data objects using name/value pairs.

JSON is best known for the curly brace syntax; every JSON object begins and ends with a curly brace. JSON builds objects using name/value pairs that are similar to the key/value pairs in Python dictionaries. Values can be strings, lists, numbers, Booleans, and nested JSON objects.

Figure 5-3 illustrates a JSON response to a **show hostname** request. The first marked line indicates the input that was received in the request. The second marked line is part of the body of the response that shows the hostname **nxosv.cisco.com**.

Figure 5-3 JSON Response Example



```
{
  “ins_api”: {
    “type”: “cli_show”,
    “version”: “1.2”,
    “sid”: “eoc”,
    “outputs”: {
      “output”: {
        “input”: “show hostname”,
        “msg”: “Success”,
        “code”: “200”,
        “body”: {
          “hostname”: “nxosv.cisco.com”
        }
      }
    }
  }
}
```

Example JSON response for a “show hostname” request

JSON is extremely popular in part because it is easy to read. In addition, but for Python programmers, it natively maps into Python as a Python dictionary. So, if you understand dictionaries, you understand JSON and vice versa. A Python dictionary is an unordered collection of items. Each item of a dictionary has a *key/value* pair. Dictionaries are optimized to retrieve values when the key is known.

Technically, a dictionary is not a JSON object. While you can print a dictionary with the **print** statement in Python, you can dump it as a JSON string by using the JSON Python module with the **dumps()** function. For example, in Example 5-11, the **print** function is used three times to illustrate the use of the **dumps()** function. The results of the **print** function are highlighted at the bottom of the example.

Example 5-11 JSON Example Using the Print and Dump Functions

```
#!/usr/bin/env python

import json

if __name__ == "__main__":
    facts = {
        'hostname': 'nxosv',
        'os': '9.3(3)
        'location': 'San_Jose'
    }
    # print facts dictionary
    print facts
    # print facts as a JSON string
    print json.dumps(facts, indent=4)
    # print a specific value of key
    print facts['os']

$ python json_test.py
{'hostname': 'nxosv', 'os': '9.3(3)', 'location': 'San_Jose'}
{
     "hostname": "nxosv",
     "os": "9.3(3)",
     "location": "San_Jose"
}
9.3(3)
```

Another function in the JSON Python module is **loads()**. It is important to understand that APIs return JSON strings, and you cannot natively work with them as dictionaries. An intermediate step is often required. This step is where the **loads()** function comes into play. Example 5-12 shows a string called **facts** being defined. Two **print** functions are executed. The third **print** function is commented to illustrate the error that it would produce if it were executed. The fourth **print** function uses the **dumps** function to treat **facts** as a dictionary.

Example 5-12 JSON Example Using the Loads and Dump Functions

```
facts = '{"hostname": "nxosv", "os": "7.3", "location": "San_Jose"}'
print facts
print type(facts)
# print facts['os']
# TypeError: string indices must be integers, not str
factsd = json.loads(facts)
print json.dumps(factsd, indent=4)

$ python json_test2.py
{"hostname": "nxosv", "os": "7.3", "location": "San_Jose"}
<type 'str'>
{
    "hostname": "nxosv",
    "os": "7.3",
    "location": "San_Jose"
}
```

Although **facts** looks like a dictionary, it is a string. You can see the quotes surrounding the dictionary. Example 5-12 is simulating an API response as a JSON string. A statement is needed to *load* the JSON string. It, in turn, creates a dictionary object that you can work with in Python, using the **dumps** function.

Cisco has been adding JSON support on various device platforms. Example 5-13 comes from a Cisco Nexus switch. As you can see in this example, you can pipe commands to JSON to see what output looks like as a JSON object. This is a great help when building Python scripts because you can see how the output from the CLI maps to the key/value pairs in Python.

Example 5-13 JSON Example Using a Cisco Nexus Switch

```
nk9v# show hostname
nk9v
nk9v# show hostname | json
{"hostname": "nk9v"}
nk9v#
```

XML

XML, like JSON, offers a way to provide structured data between computer systems. While it is not as easy as JSON for humans to understand visually, it is easy for machines to parse and generate. XML may look similar to HTML, but they are in fact different. While both use tags to define objects and elements, HTML is used to display data. A web browser knows how to display websites because it consumes an HTML object and displays it. XML, on the other hand, is used to describe data such that the XML client (for example, programming language) can consume an object that has meaning to it.

Just as you can work with JSON objects in Python, you can work with XML in Python. The main difference is that there is not a native Python data type for XML as there is for JSON. One common library that is used to work with XML in Python is **lxml**. When working with XML in Python, keep in mind that there is a distinct difference between XML strings and XML objects, just as there is a difference between a JSON string and a dictionary.

Example 5-14 shows the Python shell being launched from the Linux terminal. An XML string is defined (**xml_str**) and used to simulate a REST API response from a network device. The **etree** object is imported from the **lxml** Python module. This object is used to convert XML strings to actual objects and vice versa. The string **xml_str** is then converted to an XML object by using the **fromstring** method. The **find** method searches the full XML object for the interface object. The text is then printed.

Example 5-14 XML Example Using Python

```
cisco@cisco:~$ python
Python 2.7.11+ (default, Apr 17 2016, 14:00:29)
[GCC 5.3.1 20160413] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> xml_str = '<interfaces><interface>Gi1/1</interface></interfaces>'
>>>
>>> from lxml import etree
>>>
```

```
>>> xml_data = etree.fromstring(xml_str)
>>>
>>> intf = xml_data.find('.//interface')
>>> print intf.text
'Gi1/1'
>>>
```

XML namespaces are fairly common when using XML APIs on network devices, so it is important to understand them and to know why they are used. It is quite possible that you will never experience a conflict of XML tags when working with network devices, but you might want to create custom XML objects in an application you are writing. In doing that, you have the ability to create a namespace, which essentially becomes an identifier for each XML element. This way, you can read data from more than one source, create a larger object, and not overwrite an object, as the objects would still be accessible by their namespaces.

Nexus devices also allow you to pipe output to XML, just as with JSON. The major difference is that when you pipe output to JSON, it represents the output you will get back from NX-API. While NX-API also supports XML, when you pipe output to XML, you see the XML object that represents the NETCONF response.

Example 5-15 illustrates the use of the Cisco namespace. First, the **show hostname | xml** command is used to convert the **show hostname** command to XML format. The XML output from the CLI confirms the default Cisco NX-OS namespace, as well as the NETCONF-supported namespace. The outputted XML is then assigned to a variable called **show_hostname_xml_str**. Using the same procedure as earlier, **etree.fromstring()** and the **find** method print the hostname of the device.

Example 5-15 XML Namespaces

```
n9kv# show hostname | xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<nf:rpc-reply xmlns="http://www.cisco.com/nxos:1.0:vdc_mgr"
  xmlns:nf="urn:ietf:params:xml:ns:netconf:base:1.0">
 <nf:data>
  <show>
   <__XML__BLK_Cmd_SHOW_HOSTNAME_hostname>
    <__XML__OPT_Cmd_SHOW_HOSTNAME___readonly__>
     <__readonly__>
      <hostname>nk9v</hostname>
     </__readonly__>
    </__XML__OPT_Cmd_SHOW_HOSTNAME___readonly__>
   </__XML__BLK_Cmd_SHOW_HOSTNAME_hostname>
  </show>
 </nf:data>
</nf:rpc-reply>
```

```
]]>]]>
nk9v#

>>> show_hostname_xml_str = “””
<nf:rpc-reply xmlns="http://www.cisco.com/nxos:1.0:vdc_mgr"
  xmlns:nf="urn:ietf:params:xml:ns:netconf:base:1.0">
 <nf:data>
  <show>
   <__XML__BLK_Cmd_SHOW_HOSTNAME_hostname>
    <__XML__OPT_Cmd_SHOW_HOSTNAME___readonly__>
     <__readonly__>
      <hostname>nk9v</hostname>
     </__readonly__>
    </__XML__OPT_Cmd_SHOW_HOSTNAME___readonly__>
   </__XML__BLK_Cmd_SHOW_HOSTNAME_hostname>
  </show>
 </nf:data>
</nf:rpc-reply>
"""

>>> show_hostname_xml_data = etree.fromstring(show_hostname_xml_str)
>>> hostname = show_hostname_xml_data.find('.//{http://www.cisco.com/
  nxos:1.0:sysmgrcli}hostname')
>>>
>>> print hostname.text
nxosv
```

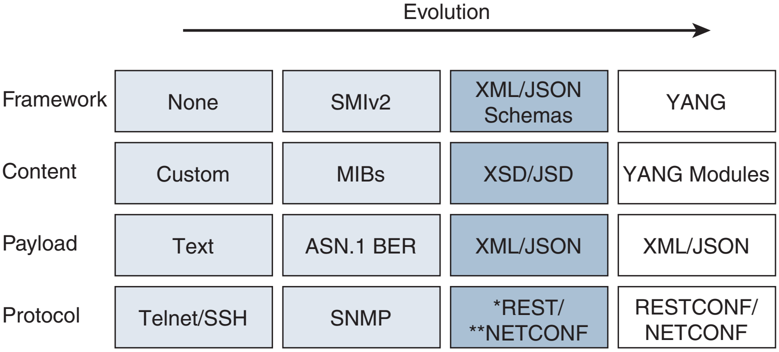
Data Models

Data models are used to describe syntax and semantics for working with specific data objects. A data model describes a constrained set of data in the form of a schema language. It uses well-defined parameters to standardize the representation of data from a network device so the output among various platforms is the same. Data models are not used to actually send information to devices but rely on protocols such as NETCONF and RESTCONF to send JSON- or XML-encoded documents. Device configuration can be validated against a data model in order to check whether the changes are valid for the device before committing the changes.

While there is a lot of focus on data models these days, they are not new. They are simply improving and becoming more focused on network devices and next-generation APIs.

As shown in Figure 5-4, the industry is migrating from a world of having no framework (no modeling) when using CLI commands and text output to a world of fully modeled devices—that is, devices that have JSON and XML representations of their full configurations and that are fully driven from a robust model such as YANG. Models also define the operational data and statistics of devices.

Figure 5-4 Evolution of Device Management Framework



Framework	None	SMIv2	XML/JSON Schemas	YANG
Content	Custom	MIBs	XSD/JSD	YANG Modules
Payload	Text	ASN.1 BER	XML/JSON	XML/JSON
Protocol	Telnet/SSH	SNMP	*REST/ **NETCONF	RESTCONF/ NETCONF

YANG

Yet Another Next Generation (YANG) is a formal contract language with rich syntax and semantics on which you can build applications. It provides these rich semantics with specific constraints and also provides reusable structures that are available within and between YANG models. YANG has been around since 2010 but has been tightly coupled with the NETCONF management protocol, as the title of its RFC suggests: "RFC 6020: YANG—A Data Modeling Language for NETCONF." Recently, YANG models are being used independently of NETCONF and for other protocols such as RESTCONF and gRPC.

There are a variety of types of YANG models:

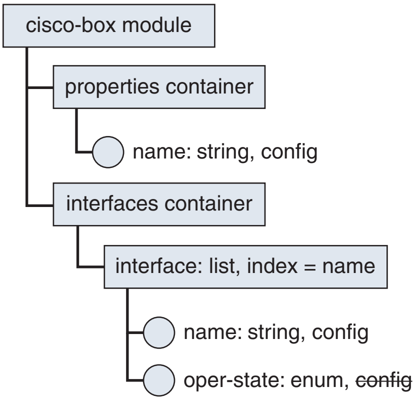
- **Industry-standard models:** These models come from various working groups, such as the IETF and the OpenConfig working group, that aim to create vendor- and platform-independent models with core features and operational statistics that are relevant across a wide variety of devices.
- **Cisco common models:** Because many features are common across Cisco devices, there are Cisco native models and native models per Cisco OS.
- **Cisco platform-specific models:** For platform- or hardware-specific features, there are additional models that are used to ensure that features mapped to a given platform are model driven so that APIs can be used for those features.

The concept of data models is not new. YANG modules are for RESTCONF/NETCONF what MIBs are for SNMP, and readability is the number-one priority for YANG.

As you can see in Figure 5-5, YANG data modeling is tree structured. The original creators of YANG wanted it to be very close to the device instrumentation. A lot of time was spent with network operators and architects to ensure that any construct in YANG reflects what is running in both standard features (common protocols) and proprietary features.

YANG is a full-blown language. For example, it is not used to express process or for actual transport. A data model describes how data is represented and accessed. In YANG, data models are represented by definition hierarchies called *schema trees*. Instances of schema trees are called *data trees*, and they are encoded in XML.

Figure 5-5 YANG Tree Structure



YANG is protocol independent, but it was originally designed specifically for NETCONF. As discussed earlier, it is also being used with REST interfaces. YANG model-driven interfaces that use REST are now called RESTCONF.

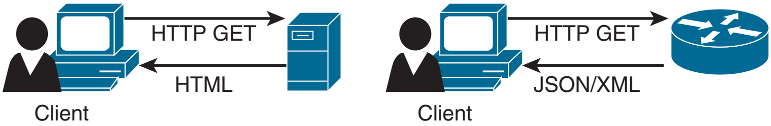
All YANG modules are publicly available. You can see the largest collection of models on GitHub, at https://github.com/YangModels/yang. This repository includes standards-based models from the Institute of Electrical and Electronics Engineers (IEEE) and Internet Engineering Task Force (IETF), as well as open-source models such as Open Daylight and vendor-specific models.

REST

There is often a perception that Representational State Transfer (REST) is a complex topic to learn about, but in reality, it is analogous to browsing a website with a web browser.

The process is very simple. First, there is a client (web browser) that performs Hypertext Transfer Protocol (HTTP) operations (such as GET) to a server (web server). The server understands the request, which is based on a given URL, and then returns HTML so that the client can properly display the HTML objects. This is illustrated on the left side of Figure 5-6.

Figure 5-6 REST HTTP Get



When using REST APIs, you have a client (with different options, including web browser plug-ins) that performs HTTP operations (for example, GET) to a network device running a web server. The network device understands the request, which is based on a uniform resource locator (URL) (and body), and returns JSON- or XML-encoded objects so that your client can properly consume them.

REST was first proposed by Roy Fielding in 2000. It was developed using simple create, retrieve, update, and delete (CRUD) operations for use with HTTP. REST uses a stateless client/server model, which means each request is treated independently from any other request. Therefore, the communication is a simple request and response. Also, REST uses simple URIs, which enables easy and familiar device programming using a web browser and graphical user interface.

CRUD operates by using the specific HTTP verbs create, retrieve, update, and delete, which are the four major functions that are implemented in database applications. You use these operations to work with network APIs to create objects (for example, create loopback interfaces), retrieve objects (particular sections of a configuration or operational data), update objects (perform changes), or delete objects (remove routes, IP addresses, and so on). These operations are used with the URL and payload so that the server (network device) knows what action to perform.

HTTP verbs are the methods that are used to perform some actions on specific resources. Because HTTP is a standardized and ubiquitous protocol, the semantics are very well known. The following are the most common HTTP verbs used by REST:

- **GET:** GET is used to read or retrieve information from a resource returning a representation of the data in JSON or XML. Because the GET method only reads data and does not change it, it is considered a "safe" method, which means there is no risk of data corruption.
- **POST:** POST creates new resources, which means it is not considered a safe method.
- **PUT:** PUT is normally used to update or replace a resource that already exists. "PUT-ing" involves sending a request with the updated representation of the original resource.
- **PATCH:** PATCH is similar in some ways to PUT in that it modifies the capabilities of a resource. The difference between PUT and PATCH is that PATCH sends a request containing only the changes to the resource and not a complete updated resource.
- **DELETE:** DELETE simply deletes a resource that is identified by a URI.

When an HTTP method is used, a specific response code is returned. For example, upon successful deletion of a resource using DELETE, the client receives a "200" message, signifying that the request succeeded.

The IETF developed the HTTP response codes, which are easy to look up online. These codes are very useful in troubleshooting because they provide specific information regarding errors on the client side or on the server side. For example, if a client receives a "400" response from a server, you can conclude that there is a syntax problem in the request.

We will take a detailed look at REST and REST APIs as they relate to Cisco DNA Center on Day 4, "REST APIs."

NETCONF

The Network Configuration Protocol (NETCONF) is an IETF network management protocol designed specifically for configuration management.

NETCONF is a next-generation network management protocol that is designed specifically for transactional-based network management and to improve on the weaknesses of SNMP. NETCONF makes a distinction between configuration and state data. The information that can be retrieved from a running system is separated into two classes: configuration data and state data. Configuration data is the set of writable data that is required to transform a system from its initial default state into its current state. State data is the additional data on a system that is not configuration data, such as read-only status information and collected statistics. To account for these issues, the NETCONF protocol recognizes the difference between configuration data and state data and provides operations for each, which are as follows:

- **Utilizes multiple configuration data stores (candidate, running, startup):** This is one of the unique attributes of NETCONF, although a device does not have to implement this feature to support the protocol. NETCONF utilizes a candidate configuration, which is simply a configuration with all proposed changes applied in an uncommitted state. It is the equivalent of entering CLI commands and having them not take effect right away. You would then commit all the changes as a single transaction, and after they are committed, you can see them in the running configuration.
- **Supports configuration change transactions:** NETCONF supports device transaction, which means that when you make an API call configuring multiple objects and one fails, the entire transaction fails, and you do not end up with a partial configuration. Changes are also configurable and platform dependent. For example, you can force a change to occur so that it fails on error or rolls back on error.
- **Provides client-side configuration validation:** This is a by-product of using YANG with NETCONF.
- **Uses filtering mechanisms for selective data retrieval:** NETCONF offers the ability to use filters that are based on XML documents or expressions and to selectively request specific components of configuration or operational state data.
- **Uses a client/server model and SSH as the transport protocol:** NETCONF is session oriented and stateful, whereas other APIs, such as native REST and RESTCONF, are stateless.

There are four core layers to the NETCONF protocol stack:

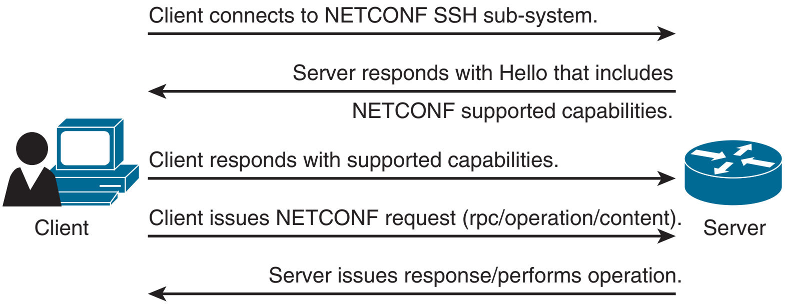
- **Protocol:** This layer specifies how the NETCONF client communicates with the NETCONF server. At times, NETCONF acts as an SSH subsystem, using the SSH protocol as the transport, although other transport types are supported per the NETCONF standard, such as TLS and Simple Object Access Protocol (SOAP).
- **Messages:** NETCONF encodes everything in XML, starting with the XML header and message. The first element in an XML document is always the RPC element, <rpc>, which tells the server that a remote procedure call (RPC) is going to be used on the device. The RPCs map directly to specific operations on the device.
- **Operations:** Each device and platform supports a given number of operations. These are some of the common ones:
    - **<get>:** Retrieves running configuration and device state information
    - **<get-config>:** Retrieves all or part of a configuration datastore

  - **<edit-config>:** Loads all or part of a configuration to the specified datastore
  - **<delete-config>:** Deletes a configuration datastore
  - **<copy-config>:** Replaces an entire configuration datastore with another
  - **<lock> and <unlock>:** Locks or unlocks the entire configuration datastore system
  - **<close-session>:** Gracefully terminates a NETCONF session
  - **<kill-session>:** Forces termination of a NETCONF session
- **Content**: Embedded as XML objects within the operations tag is the data you want to retrieve or configure. It is the content that is an XML representation of YANG models or XML Schema definitions.

NETCONF over SSH

The steps that occur during a NETCONF session are illustrated in Figure 5-7.

Figure 5-7 NETCONF over SSH


Client connects to NETCONF SSH sub-system.
Server responds with Hello that includes NETCONF supported capabilities.
Client responds with supported capabilities.
Client issues NETCONF request (rpc/operation/content).
Server issues response/performs operation.
Client
Server

Example 5-16 shows these steps as a client device establishes an SSH session with a Cisco CSR 1000V router. Notice that the NETCONF subsystem is specified in the SSH command, and the client is connecting over port 830, which is the default port for NETCONF. The router (server) then sends a hello that includes all the router's supported NETCONF capabilities. When the server sends its hello, the client needs to send a hello with its supported capabilities. The client can respond back with everything the server supports (assuming the client does, too) or just with the bare minimum to do edits and gets. The example shows that the client just supports NETCONF base.

After the client sends its capabilities, it can then send NETCONF requests. In Example 5-16, the request being sent is performing the NETCONF get operation (denoted by <get>). It then asks for a given section of configuration by using a filter. Based on the devices/servers being used, you often have to be aware of which model is being used (denoted by the XML namespace). The filter in this example is selectively asking for configuration information of GigabitEthernet1. The router responds with the configuration of GigabitEthernet1. Then the server processes the client's request and responds with the configuration, as expected.

Note that we have added solid lines to this example to separate the phases of the process.

Example 5-16 NETCONF over SSH

```
$ ssh -p 830 cisco@csr1kv -s netconf
cisco@csr1kv's password:
```

```
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
<capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:writable-running:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:xpath:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:validate:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:rollback-on-error:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:notification:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:interleave:1.0</capability>
    <capability>http://tail-f.com/ns/netconf/actions/1.0</capability>
    <capability>http://cisco.com/ns/yang/ned/ios?module=ned&amp;revision=2016-06-20
</capability>
    <omitted for brevity>
</capabilities>
<session-id>203499</session-id></hello>]]>]]>
```

```
 <?xml version="1.0"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:xml:ns:netconf:base:1.0</capability>
  </capabilities>
</hello>]]>]]>
```

```
 <?xml version="1.0"?>
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <get>
        <filter type="subtree">
            <native xmlns="http://cisco.com/ns/yang/ned/ios">
             <interface>
              <GigabitEthernet>
               <name>1</name>
              </GigabitEthernet>
             </interface>
            </native>
        </filter>
    </get>
```

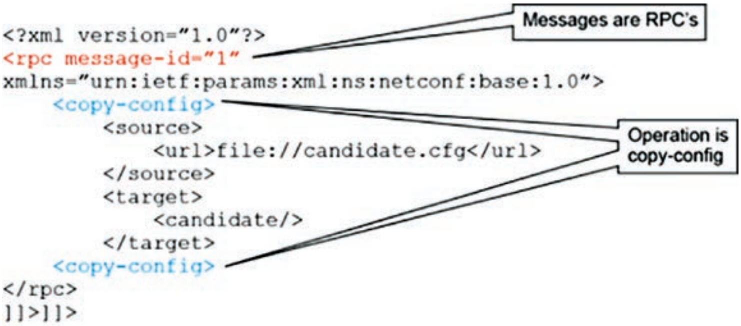
```
</rpc>
]]>]]>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-
  id="101"><data><native xmlns="http://cisco.com/ns/yang/ned/ios"><interface>
  <GigabitEthernet><name>1</name><negotiation><auto>true</auto></negotiation>
  <ip><address><primary><address>192.168.1.11</address><mask>255.255.255.0</mask>
  </primary></address></ip></GigabitEthernet></interface></native></data>
  </rpc-reply>]]>]]>
```

XML Encoding

In NETCONF requests, you can map the message to its value, the operation to its value, and the content to the value. The message is always <rpc>. Within a message, the first element is the NETCONF operation, such as <copy-config> or <get>, and then within the message is content that describes what action should be taken for the given operation. Figure 5-8 shows the RPC message ID 1 and the operation copy-config.

Figure 5-8 NETCONF copy-config

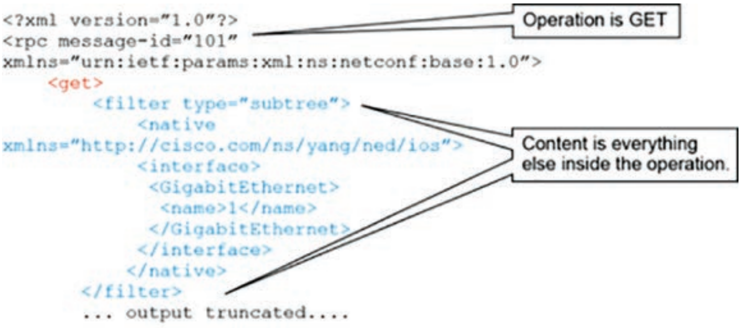


```
<?xml version="1.0"?>
<rpc message-id="1"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <copy-config>
        <source>
            <url>file://candidate.cfg</url>
        </source>
        <target>
            <candidate/>
        </target>
    <copy-config>
</rpc>
]]>]]>
```

Messages are RPC's
Operation is copy-config

Figure 5-9 shows a get operation. Everything else inside this operation is content.

Figure 5-9 NETCONF Get



```
<?xml version="1.0"?>
<rpc message-id="101"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <get>
        <filter type="subtree">
            <native
xmlns="http://cisco.com/ns/yang/ned/ios">
            <interface>
             <GigabitEthernet>
              <name>1</name>
             </GigabitEthernet>
            </interface>
           </native>
       </filter>
       ... output truncated....
```

Operation is GET
Content is everything else inside the operation.

NETCONF Configuration Datastores

NETCONF defines the existence of one or more configuration datastores and allows configuration operations on those datastores. A configuration datastore is a complete set of configuration data that is required to get a device from its initial default state into a desired operational state. A configuration datastore does not include state data or executive commands.

The existence of different datastores on a given device is advertised via *capabilities*, as defined in RFC 6241. As mentioned earlier today, capabilities are exchanged when you start a NETCONF session with a device, and you can see these capabilities in the initial message from the NETCONF server on the device.

The running configuration datastore holds the complete configuration that is currently active on a network device. Only one running configuration datastore exists on a device, and it is always present. NETCONF protocol operations refer to this datastore using the <running> XML element. The running datastore capability is supported only in Cisco IOS XE and Cisco NX-OS Software.

Cisco IOS XR Software supports a candidate datastore. When using NETCONF with Cisco IOS XR Software, you must explicitly state that it is the candidate datastore that is the target of operations, and you must explicitly commit any changes.

RESTCONF

Using REST APIs that are based on HTTP and JSON is becoming a common way of programming infrastructure. RESTCONF is a specification for a REST-like interface that maps to the NETCONF/YANG interface model. RESTCONF is basically NETCONF/YANG without the SSH but with JSON and HTTP(S).

RESTCONF is a REST-like protocol that provides an HTTP-based mechanism for accessing data that is defined in NETCONF datastores and modeled in YANG. RESTCONF combines the simplicity of HTTP with the predictability and automation potential of a schema-driven API. A client can determine all management resources for YANG models and NETCONF capabilities, which means the URIs for custom protocol operations and datastore content are predictable, based on the YANG module definitions. The generation of the code to support RESTCONF APIs and the mapping of those API calls to NETCONF can be automated because the mapping from a YANG model to a RESTCONF URI is well defined.

RESTCONF helps support a common, REST-based programming model for network programming in general. This model aligns with the wider trend in infrastructure programming to support REST APIs.

RESTCONF Methods

RESTCONF maps HTTP methods to specific semantics in the RESTCONF protocol:

- **GET:** The GET method is sent by the client to retrieve data and metadata for a resource. It is supported for all resource types except for operation resources. The HTTP GET operation uses the same semantics as the NETCONF get and get-config operations and can also be used for notifications.

- **PATCH:** The PATCH method uses the HTTP PATCH method to provide a framework for resource patching mechanisms. It is basically the equivalent of the NETCONF operation=merge. The HTTP PATCH operation supports partial configuration updates in a way that is similar to the NETCONF edit-config operation with operation=merge.
- **PUT:** The client sends the PUT method to create or replace the target resource. The HTTP PUT operation is similar to PATCH, but it is typically used to replace the contents of a named resource in lieu of changing resource attribute values.
- **POST:** The client sends the POST method to create a data resource or invoke an operation resource. The HTTP POST operation is used for NETCONF RPCs and, in some circumstances, to create a resource.
- **DELETE:** The DELETE method is used to delete the target resource. The HTTP DELETE operation is equivalent to the NETCONF edit-config with operation=delete.

This mapping follows the general pattern of most REST APIs. Resources representing configuration data can be modified with the DELETE, PATCH, POST, and PUT methods. Data is encoded with either XML or JSON.

The client can also access the YANG libraries (that is, the capabilities) that the server implements.

The API resource contains the entry points for the RESTCONF datastore and operation resources. It is the top-level resource that is referred to with the notation {+restconf} and has the media type application/yang.api+xml or application/yang.api+json, depending on whether the encoding of the payload document is XML or JSON.

Note that RESTCONF does not support all the NETCONF operations. Specifically, it does not support the following operations:

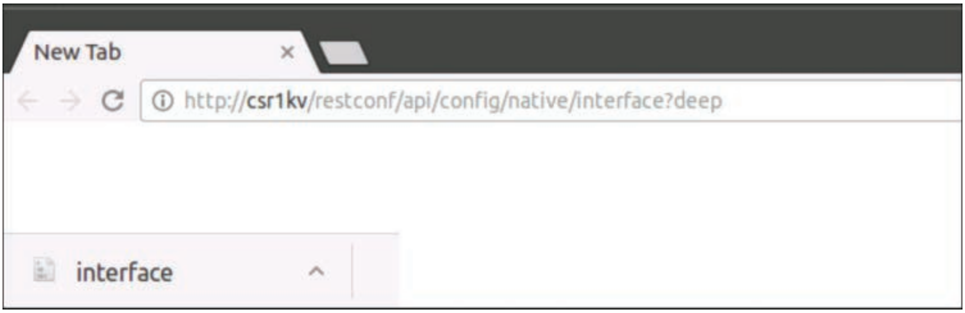
- Config locking
- Candidate config
- Startup config
- Validate
- Confirmed commit

RESTCONF GET Example

As with NETCONF, a very common use case for RESTCONF is getting portions of the running configuration datastore. With RESTCONF, you can get portions of the running configuration datastore with HTTP GET requests and view the result. The simplest way is to use a browser and access the specific RESTCONF URL.

Figure 5-10 and Example 5-17 use a CSR 1000V router running IOS XE Version 16.9.5 and show a user making a GET request via a browser to view the router's interfaces. Notice that the browser URL specifies the **deep** query parameter for the **interface** resource. This produces more detailed output of all parameters associated with all the interfaces on the router. Without this option, you would see only the interface identifiers.

Figure 5-10 RESTCONF Get on IOS XE



Once the URL is entered, the router responds with a text file called interface. (Example 5-17 shows the contents of this file edited for brevity because the output produced by this particular get command is quite long.)

Example 5-17 RESTCONF Example

```
<interface . . .>
  <GigabitEthernet>
    <name>1</name>
    <switch>
    </switch>
    <switchport>
      <access>
      </access>
      <trunk>
       <allowed>
<. . .output omitted. . .>
    <ip>
      <address>
        <primary>
          <address>192.168.1.11</address>
          <mask>255.255.255.0</mask>
<. . .output omitted. . .>
```

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	28
CCNP Enterprise Design ENSLD 300-420 Official Cert Guide: Designing Cisco Enterprise Networks	12

Day 4

REST APIs

ENCOR 350-401 Exam Topics

Security

- Describe REST API security

Automation

- Describe APIs for Cisco DNA Center and vManage
- Interpret REST API response codes and results in payload using Cisco DNA Center and RESTCONF

Key Topics

Today we review how Cisco DNA Center and Cisco vManage use REST APIs for communication. We also look at how to interpret REST API response codes and secure communication between the controller and the application or network devices.

As an enterprise network grows, traditional management becomes more challenging. Advanced systems, such as Cisco DNA Center, can be too complex to be easily managed using traditional tools. To meet these challenges, systems can utilize application programming interfaces (APIs) for configuration and monitoring within the system and with external management utilities.

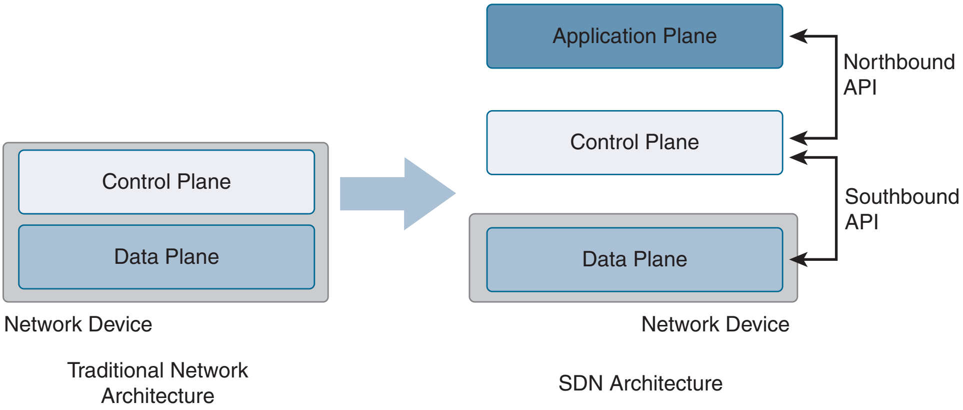
Application Programming Interfaces

Generally speaking, an API is code that enables two different software programs to communicate with each other. The software-defined networking (SDN) architecture is composed of three stacked layers, as illustrated in Figure 4-1:

- **Data (or forwarding) plane:** This layer contains network elements (that is, any physical or virtual devices that deal with customer traffic).
- **Control plane:** This is the core layer of the SDN architecture. It contains SDN controllers, which provide centralized control of the devices in the data plane.
- **Application plane:** This layer contains the SDN applications that communicate their network requirements to the controller.

An SDN controller uses APIs to communicate with the application and data planes. Communication with the data plane involves southbound APIs, and services are offered to the application plane using the northbound APIs.

Figure 4-1 SDN Three-Layer Architecture



Southbound APIs

Southbound APIs (or device-to-control plane interfaces) are used for communication between controllers and (physical) network devices. The following are some commonly used southbound APIs:

- **OpenFlow:** This industry-standard API is defined by the Open Networking Foundation (ONF). It configures white-label switches and, as a result, defines the flow path through the network. The actual configuration of the devices is accomplished with the use of NETCONF.
- **NETCONF:** The IETF standardized this network management protocol, which provides mechanisms to install, manipulate, and delete the configuration of network devices via remote procedure call (RPC) mechanisms. The messages are encoded using XML. Not all devices support NETCONF; the ones that do support it advertise their capabilities via the interface.
- **OpFlex:** This open-standard protocol provides a distributed control system that is based on a declarative policy information model. The big difference between OpFlex and OpenFlow lies in their respective SDN models. OpenFlow uses an imperative SDN model, where a centralized controller sends detailed and complex instructions to the control plane of the network elements to implement a new application policy. In contrast, OpFlex uses a declarative SDN model. The controller, which is a Cisco Application Policy Infrastructure Controller (APIC), sends a more abstract policy to the network elements. The controller trusts the network elements to implement the required changes by using their own control planes.

Northbound APIs

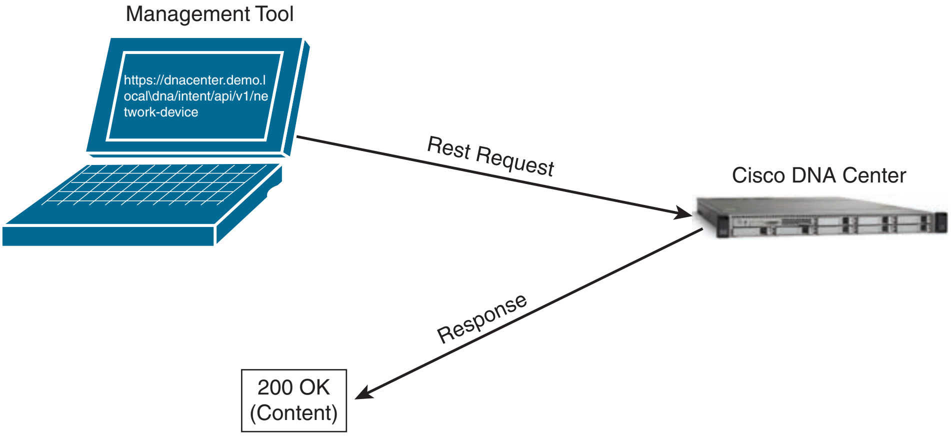
Northbound APIs and northbound interfaces are responsible for the communication between the SDN controller and the services that run over the network. Northbound APIs enable applications to manage and control the network. So rather than adjusting and tweaking your network repeatedly to get a service or an application running correctly, you can set up a framework that allows the application to demand the network setup that it needs. These applications range from network virtualization and dynamic virtual network provisioning to more granular firewall monitoring, user identity management, and access policy control.

Unfortunately, currently there is no single northbound interface that you can use for communication between a controller and all applications. Instead, you use various different northbound APIs, each working only with a specific set of applications.

REST API Response Codes and Results

As mentioned on Day 5, "Network Programmability," Representational State Transfer (REST) is based on the HTTP request-response model, as illustrated in Figure 4-2.

Figure 4-2 REST Communication Process



The Cisco DNA Center Intent API uses the GET, POST, PUT, and DELETE HTTP actions only:

- **GET:** Retrieves data from the path that the URL specifies.
- **POST:** Writes new data at the path that the URL specifies. Used only to create new data on the server.
- **PUT:** Replaces existing data at the URL path. Cannot be used to create new data, only to update existing data.
- **DELETE:** Removes existing data at the URL path.

REST uses HTTP or HTTPS to send a request to a web service, which returns a response. One part of the REST request is similar to the text that you type into your web browser's address bar to go to a website. This text is a uniform resource locator (URL), and it is part of the request that your browser sends to the web service to ask that service to do something. The URL specifies the location of the resource that the web server uses to fulfill the request. Another part of the request specifies the action that the web service is to perform using the specified resource. Common requests include getting data from the resource that the URL specifies and writing data to the resource that the URL specifies. For example, when you click on a hypertext link, your browser sends a request to get the resource that is located at the path the URL specifies.

The GET, POST, PUT, and DELETE actions above are sometimes referred to as *CRUD operations*, for create, read, update, and delete.

HTTP Status Codes

When a web service processes an HTTP or REST request, it returns a standard HTTP status code that indicates whether the request succeeded.

If a request is successful, the web service returns a 2*xx* response. The following are two common 2*xx* response codes:

- **200 (OK):** The request succeeded, and the response includes the result of the action you requested. For example, if you tried to create a user account, the status value 200 means that the account was created successfully.
- **202 (Accepted):** The server accepted your request, but it must perform other actions to complete the request. For example, if Cisco DNA Center needs to interact with network devices to fulfill the request, it returns a 202 status code.

Status codes may also be used to indicate that a request did not complete successfully. If an error was caused by a client request, the server returns a 4*xx* status code. For example, the status code 401 (Unauthorized) means that the REST caller does not have sufficient privileges to complete the request.

If an error was caused by a server error, the server returns a 5*xx* status code. For example, the status code 503 (Service Unavailable) means the server was unavailable to fulfill the request at the time of the request.

Although there are many HTTP status codes, the DNA Center Intent API uses only a handful of them; the other codes are intended for use by web services that interact with web browsers. The pi reference document describes the meaning of each HTTP status code that a particular request path returns. For example, you can find Intent APIs and HTTP response codes at https://developer.cisco.com/docs/dna-center/api/1-3-1-x/ for Cisco DNA Center version 1.3.1.x.

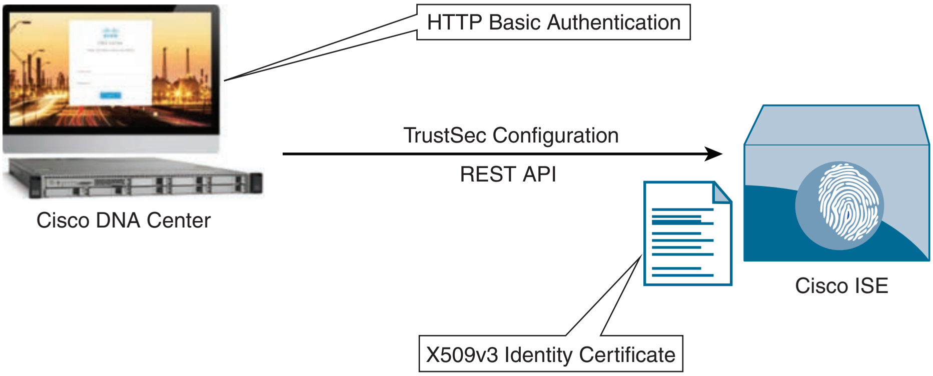
REST API Security

There are two parts to securing REST API communications. The first is securing the communication channel from prying eyes. The second is verifying that the data is from a trusted source.

In Figure 4-3, Cisco DNA Center is using a REST API to push a TrustSec configuration to Cisco Identity Service Engine (ISE). For this communication, Cisco ISE acts as the HTTPS server. Communication between Cisco DNA Center and Cisco ISE will be encrypted using Transport Layer Security (TLS), ensuring that if the data flow is captured, the data cannot be easily viewed by prying eyes.

As another security feature, Cisco ISE, as the server, provides an X509v3 identity certificate prior to secured communications with HTTPS. Cisco DNA Center must verify the identity certificate by validating the signature attached to the certificate before the secure TLS tunnel can be established. This can be done by installing the trusted root certificate of the root CA that signed the certificate or, in the case of a self-signed certificate, installing the identity certificate itself. The identity certificate ensures that Cisco DNA Center is pushing configuration to the trusted device.

Figure 4-3 Securing REST API Communication



While Cisco DNA Center can verify the identity of Cisco ISE based on the identity certificate, it is imperative that Cisco ISE verify the identity of Cisco DNA Center before applying any configuration it receives. REST API HTTPS requests are authenticated using HTTP basic authentication. The Basic Authentication header must be added to every API request, and authentication is performed for every request. For this type of authentication to work, an administrative user must be created in Cisco ISE, with permissions to access the REST API interface. For other network devices, such as the Cisco Adaptive Security Appliance (ASA), user privilege levels can restrict what features can be accessed using REST APIs.

There are other methods for securing REST APIs. For example, API keys can be used for authentication. These keys can be used to track an account's usage of an API, which makes it possible to charge by volume and features. An API key is a unique string that serves as the access credential for an API. These keys are often created manually using a web interface to the service that is being queried.

An alternative form of authentication for REST APIs is tokens. Tokens are typically used by client-side apps and issued by the server. Token authentication differs from cookie-based session management in that it is typically stateless, which means you do not need to store session details on the server. This makes it easier to scale out backend systems using cloud infrastructure, which is exactly what REST APIs are suitable for.

REST APIs in Cisco DNA Center

Cisco DNA Center is at the heart of Cisco intent-based networking. Cisco DNA Center supports the expression of business intent for network use cases such as base automation capabilities in the enterprise network.

The Analytics and Assurance features of Cisco DNA Center provide end-to-end visibility into the network, with full context through data and insights. Cisco customers and partners can use the Cisco DNA Center platform to create value-added applications that use the native capabilities of Cisco DNA Center. You can use Cisco DNA Center Intent APIs, integration flows, events and notification services, and the optional Cisco DNA Center Multivendor SDK to enhance the overall network experience by optimizing end-to-end IT processes, reducing total cost of ownership (TCO), and developing new value-added networks.

Intent API

The Cisco Intent API is a northbound REST API that exposes specific capabilities of the Cisco DNA Center platform. The Intent API provides policy-based abstraction of business intent, allowing you to focus on an outcome to achieve instead of struggling with the mechanisms that implement that outcome.

The RESTful Cisco DNA Center Intent API lets you use HTTPS verbs (GET, POST, PUT, and DELETE) and JSON syntax to discover and control your network devices centrally.

Know Your Network

Know Your Network REST request paths retrieve details about clients, sites, topology, and devices. They can also add devices to the network and export device data. The Developer Toolkit > APIs page organizes Know Your Network request paths according to the following subcategories:

- **Sites:** The Site Hierarchy Intent API retrieves the site hierarchy with network health information.
- **Networks:** The Network Health Intent API retrieves network devices by category, with health information on each of the devices returned. Additional request paths retrieve physical and virtual topologies.
- **Devices:** The Network Device Detail Intent API retrieves detailed information about devices retrieved, including timestamp, MAC address, UUID, name, and device name. Additional REST request paths allow you to retrieve additional information, such as functional capabilities, interfaces, device configuration, certificate validation status, values of specified fields, modules, and VLAN data associated with specified interfaces. You can also add, delete, update, or sync specified devices.
- **Clients:** The Client Health Intent API returns overall client health, organized into wired and wireless categories. The Client Detail Intent API returns detailed information (including health) about a single client.

Site Management

The site management APIs assist in provisioning enterprise networks with zero-touch deployment. These APIs also enable you to manage the activation and distribution of software images in the network. The following are two examples of site management APIs:

- **Software Image Management (SWIM) API:** This API enables you to retrieve information about available software images, import images into Cisco DNA Center, distribute images to network devices, and activate images that have been installed on devices.
- **Plug and Play (PnP) API:** This API enables you to manage PnP projects, settings, workflows, virtual accounts, and PnP-managed devices.

Operational Tools

Operational tools enable you to configure and manage CLI templates, discover network devices, configure network settings, and trace paths through the network. The following are examples of operational tools:

- **Command Runner API:** This API enables you to retrieve the keywords of all command-line interfaces (CLIs) that Command Runner accepts, and it lets you run read-only commands on a network device to retrieve the device's real-time configuration.
- **Network Discovery API:** This API provides programmatic access to the Discovery functionality of Cisco DNA Center. You can use this API to create, update, delete, and manage discoveries and their associated credentials. You can also use this API to retrieve the network devices that a particular discovery job acquired.
- **Template Programmer API:** This API enables you to perform CRUD (create, read, update, and delete) operations on templates and projects that the template programmer uses to facilitate design and provisioning workflows in Cisco DNA Center. You can use this API to create, view, edit, delete, and version templates. You can also add interactive commands to templates, check the contents of templates for syntactical errors or blacklisted commands, deploy templates, and check the status of template deployments.
- **Path Trace API:** This API simplifies resolution of network performance issues by tracing application paths through the network and providing statistics for each hop along the path. You can use this API to analyze the flow between two endpoints on the network, retrieve the results of a previous flow analysis, summarize all stored flow analyses, or delete a saved flow analysis.
- **Task API:** This API queries Cisco DNA Center for more information about a specific task that your RESTful request initiated. Network actions may take several seconds or minutes to complete, so Cisco DNA Center completes most tasks asynchronously. You can use the Task API to determine whether a task completed successfully; if it did, you can then retrieve more information from the task itself, such as a list of devices that have been provisioned.
- **File API:** This API enables you to retrieve files from Cisco DNA Center. For example, you might use this API to get a software image or a digital certificate from Cisco DNA Center.

Authentication

All Cisco DNA Center platform REST requests require proof of identity. The Authentication API generates a security token that encapsulates the privileges of an authenticated REST caller. Cisco DNA Center authorizes each requested operation according to the access privileges associated with the security token that accompanies the request.

Multivendor Support

Cisco DNA Center now allows customers to manage their non-Cisco devices. Multivendor support is available in Cisco DNA Center thanks to an SDK that can be used to create device packages for third-party devices. A device package enables Cisco DNA Center to communicate to third-party devices by mapping Cisco DNA Center features to those devices' southbound protocols.

Multivendor support capabilities are based on southbound interfaces. These interfaces interact directly with network devices through the CLI, SNMP, or NETCONF. Southbound interfaces are not exposed to the consumer. Instead, a consumer uses the Intent API, which abstracts the underlying complexity of the traditional network. The user of the Intent API need not be concerned with the particular protocols that the southbound interfaces use to implement network intent on devices that Cisco DNA Center supports.

Integration API

Integration capabilities are part of westbound interfaces. To meet the need to scale and accelerate operations in modern data centers, IT operators require intelligent, end-to-end workflows built with open APIs. The Cisco DNA Center platform provides mechanisms for integrating Cisco DNA Assurance workflows and data with third-party IT service management (ITSM) solutions.

ITSM minimizes handoffs, reduces duplication of issues, and optimizes processes by integrating the Cisco DNA Center platform into incident-management, change-management, and problem-management systems. It also integrates the Cisco DNA Center platform into approval and preapproval chains, and it links the Cisco DNA Center platform with formal change- and maintenance-window schedules.

The platform also integrates with reporting and analytics capabilities for capacity planning, asset management, compliance control, and auditing. The Cisco DNA Center platform boosts IT efficiency and automation by integrating seamlessly with other IT systems through RESTful APIs.

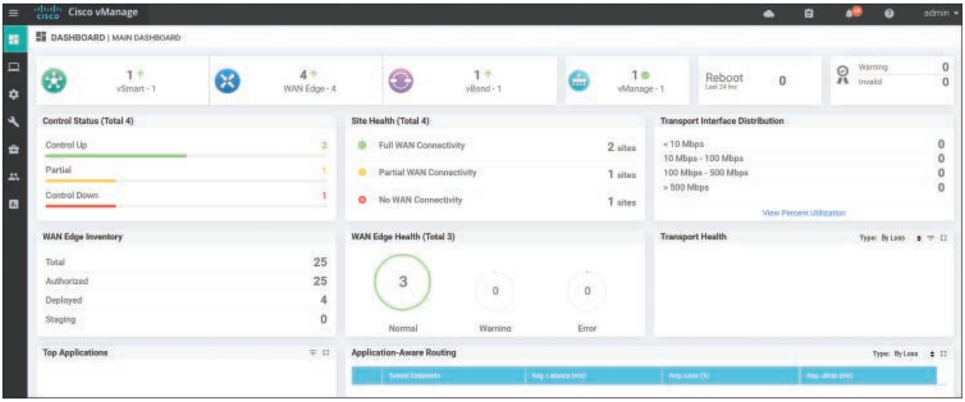
The Cisco DNA Center platform supports the IT4IT reference architecture, including the standards for events, incidents, problems, and requests for changes. If you are using an ITSM system that supports this reference architecture, Cisco DNA Center can integrate with your system.

REST APIs in Cisco vManage

Cisco Software-Defined WAN (SD-WAN) uses APIs for communication and to implement policy changes within its fabric. You access the REST API through the vManage web server.

Figure 4-4 shows the main dashboard for Cisco vManage. (All vManage screen captures and examples in this section are taken using the Cisco DevNet Reservation SD-WAN Sandbox Version 19.2 at https://devnetsandbox.cisco.com.)

Figure 4-4 Cisco SD-WAN vManage Main Dashboard GUI


Cisco vManage
admin
DASHBOARD | MAIN DASHBOARD
1 vSmart - 1
4 WAN Edge - 4
1 vBond - 1
1 vManage - 1
Reboot Last 24 hrs 0
Warning 0
Invalid 0
Control Status (Total 4)
Control Up 2
Partial 1
Control Down 1
Site Health (Total 4)
Full WAN Connectivity 2 sites
Partial WAN Connectivity 1 sites
No WAN Connectivity 1 sites
Transport Interface Distribution
< 10 Mbps 0
10 Mbps - 100 Mbps 0
100 Mbps - 500 Mbps 0
> 500 Mbps 0
View Percent Utilization
WAN Edge Inventory
Total 25
Authorized 25
Deployed 4
Staging 0
WAN Edge Health (Total 3)
3 Normal
0 Warning
0 Error
Transport Health
Type: By Loss
Top Applications
Application-Aware Routing
Type: By Loss
Tunnel Endpoints
Avg. Latency (ms)
Avg. Loss (%)
Avg. Jitter (ms)

The Cisco SD-WAN vManage REST API calls expose the functionality of SD-WAN software and hardware features and of the normal operations you perform to maintain SD-WAN devices and the overlay network. In REST API terminology, each of these features or operations is called a *resource*. A resource is an object with a type, associated data, relationships to other resources, and a set of methods that operate on it.

Resources are grouped into *collections*. Each collection contains a single type of resource and so is homogeneous. In the Cisco SD-WAN REST API, the collection of resources is present at the top level of the API. Cisco SD-WAN REST API resources are grouped into the collections shown in Table 4-1.

Table 4-1 Cisco SD-WAN Resource Collections

Resource Collection	Description
Administration	Provides resources for managing users and user groups, viewing audit logs, and managing the local vManage server.
Certificate Management	Provides resources for managing certificates and security keys.
Configuration	Provides resources for creating feature and device configuration templates, retrieving the configurations in existing templates, and creating and configuring vManage clusters.
Device Inventory	Provides resources for collecting device inventory information, including serial numbers and system status.
Monitoring	Provides resources for viewing status, statistics, and other information about operational devices in the overlay network. SD-WAN devices collect monitoring information about themselves every 10 minutes. After collecting these statistics, each SD-WAN device places the data into a zip file. The vManage server retrieves these zip files every 10 minutes or, if the vManage server cannot log in to the device, it retrieves them whenever it is next able to log in.
Real-Time Monitoring	Provides resources for retrieving, viewing, and managing real-time statistics and traffic information. Real-time monitoring information is gathered in real time, approximately once per second.
Troubleshooting Tools	Provides resources for troubleshooting devices, for determining the effect of policy, for updating software, and for retrieving software version information.

Resource Data

The Cisco SD-WAN REST API uses the JavaScript Object Notation (JSON) data model to represent the data associated with a resource. JSON uses three types of data:

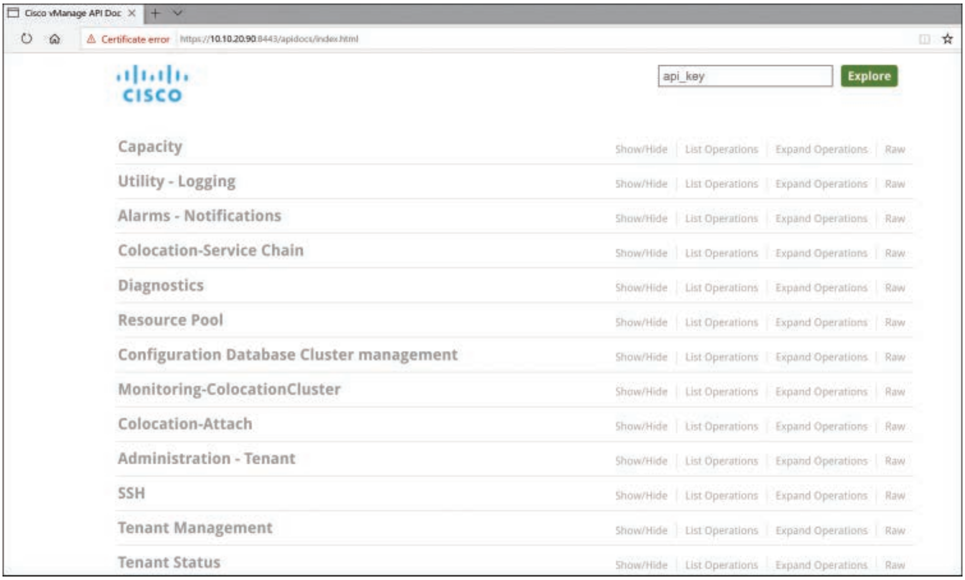
- **Scalar:** Data types with a single value each. Scalar types include numbers, strings, and Booleans.
- **Array:** Ordered lists of values of an arbitrary type.
- **Object:** Unordered sets of key/value pairs (also referred to as attributes), where key is a string and the value can have an arbitrary type.

Cisco SD-WAN API Library and Documentation

A common REST principle is that APIs should be self-descriptive and self-documenting. The resource collections and resources in the Cisco SD-WAN REST API are self-documenting, describing the data you need to make a call and the response from each call. However, the collections and resources assume that you are familiar with the Cisco SD-WAN overlay network concepts, software and hardware features, and capability.

The vManage REST API library and documentation are bundled with and installed on the vManage web application software. To access the API documentation from a web browser, use the URL https://*ip-address*:*port*/*apidocs*, where *ip-address* is the IP address of the vManage server, and *port* is the port used for the vManage server, which is typically 8443. For example, if the vManage web server has the URL https://10.10.20.90:8443, you access the vManage REST API at https://192.168.48.83:8443/apidocs. Figure 4-5 shows an example.

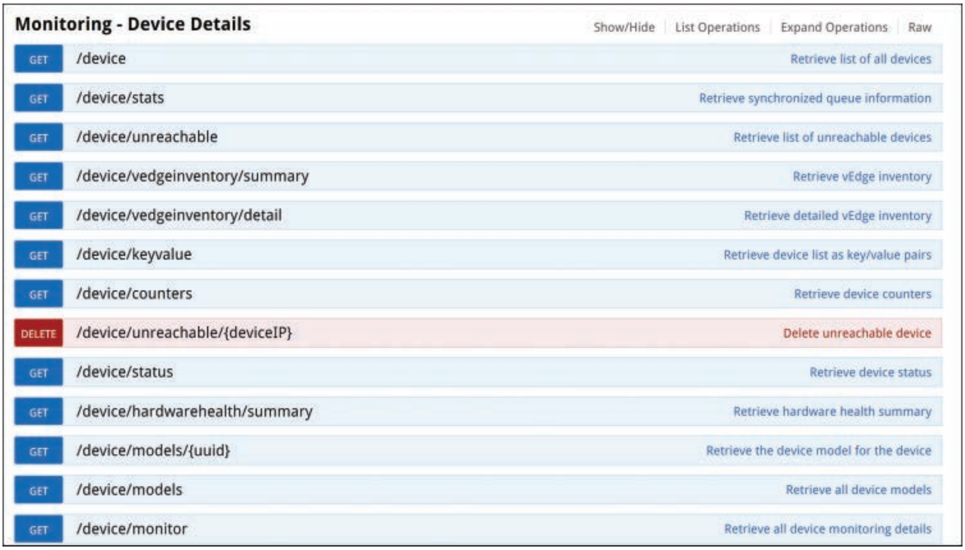
Figure 4-5 Cisco SD-WAN vManage Built-in API Library


Cisco vManage API Doc
Certificate error
https://10.10.20.90:8443/apidocs/index.html
api_key
Explore
Capacity
Utility - Logging
Alarms - Notifications
Colocation-Service Chain
Diagnostics
Resource Pool
Configuration Database Cluster management
Monitoring-ColocationCluster
Colocation-Attach
Administration - Tenant
SSH
Tenant Management
Tenant Status
Show/Hide | List Operations | Expand Operations | Raw

If you scroll down the list of APIs, you can select Monitoring—Device Details to see a list of all device-monitoring APIs supported in vManage (see Figure 4-6). In this list, different colors are used for GET and DELETE operations. GET operations are always shown in blue, DELETE operations are shown in red, POST operations are shown in green, and PUT operations are shown in brown.

You can select the /device GET operation, as shown in Figure 4-7, to expand that block of information and see the possible response messages: 200 (Success), 400 (Bad Request), 403 (Forbidden), and 500 (Internal Server Error).

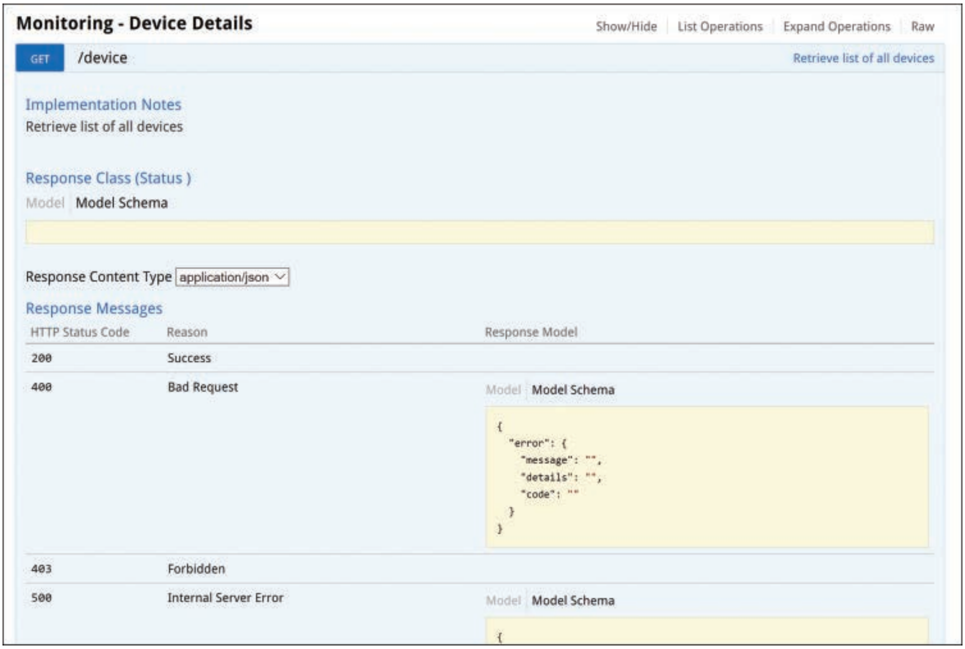
Figure 4-6 Cisco SD-WAN vManage Monitoring: Device Details API Library



Monitoring - Device Details

GET	/device	Retrieve list of all devices
GET	/device/stats	Retrieve synchronized queue information
GET	/device/unreachable	Retrieve list of unreachable devices
GET	/device/vedgeinventory/summary	Retrieve vEdge inventory
GET	/device/vedgeinventory/detail	Retrieve detailed vEdge inventory
GET	/device/keyvalue	Retrieve device list as key/value pairs
GET	/device/counters	Retrieve device counters
DELETE	/device/unreachable/{deviceIP}	Delete unreachable device
GET	/device/status	Retrieve device status
GET	/device/hardwarehealth/summary	Retrieve hardware health summary
GET	/device/models/{uuid}	Retrieve the device model for the device
GET	/device/models	Retrieve all device models
GET	/device/monitor	Retrieve all device monitoring details

Figure 4-7 Cisco SD-WAN vManage Monitoring: Device Details GET /device API



Monitoring - Device Details

GET /device Retrieve list of all devices

Implementation Notes

Retrieve list of all devices

Response Class (Status)

Response Content Type application/json

Response Messages

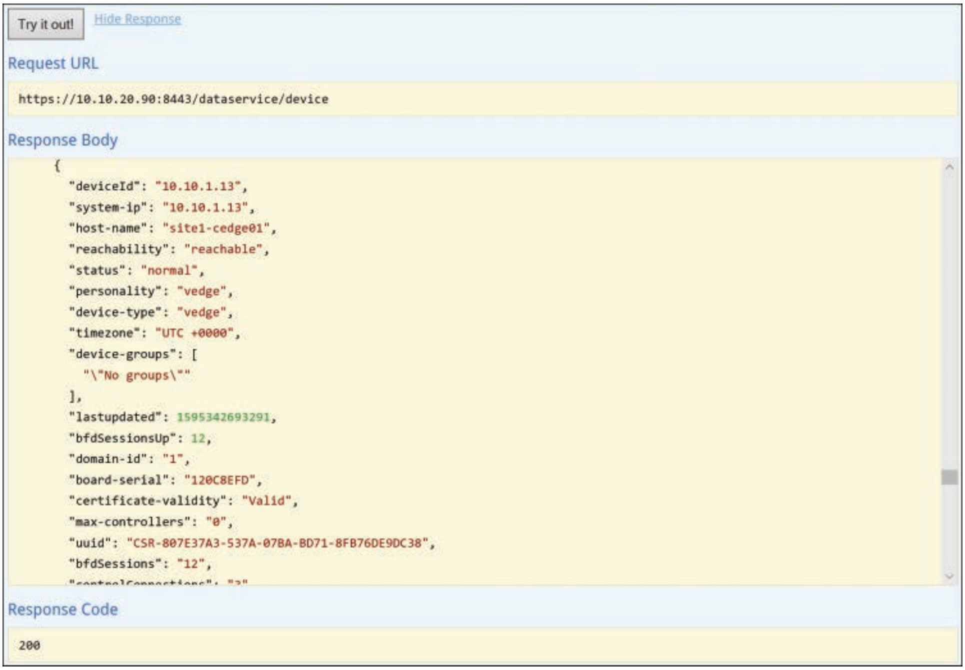
HTTP Status Code	Reason	Response Model
200	Success	
400	Bad Request	Model Schema
403	Forbidden	
500	Internal Server Error	Model Schema

```
{
  "error": {
    "message": "",
    "details": "",
    "code": ""
  }
}
```

```
{
```

If you scroll down, you can click the Try It Out! button to test a specific API. Notice that in this example, the request URL is https://10.10.20.90:8443/dataservice/device. The response body is quite long, but you can scroll through it you to locate the different Cisco SD-WAN devices deployed in the network. Figure 4-8 shows a vEdge device with IP address 10.10.1.13 deployed at Site 1. At the bottom of the figure, notice the 200 HTTP response code, which indicates that the request was processed successfully.

Figure 4-8 Cisco SD-WAN vManage Monitoring: Device Details Try It Out Function



```
https://10.10.20.90:8443/dataservice/device
```

```
    {
      "deviceId": "10.10.1.13",
      "system-ip": "10.10.1.13",
      "host-name": "site1-cedge01",
      "reachability": "reachable",
      "status": "normal",
      "personality": "vedge",
      "device-type": "vedge",
      "timezone": "UTC +0000",
      "device-groups": [
        "\"No groups\""
      ],
      "lastupdated": 1595342693291,
      "bfdSessionsUp": 12,
      "domain-id": "1",
      "board-serial": "120C8EFD",
      "certificate-validity": "Valid",
      "max-controllers": "0",
      "uuid": "CSR-807E37A3-537A-07BA-BD71-8FB76DE9DC38",
      "bfdSessions": "12",
```

```
200
```

Performing REST API Operations on a vManage Web Server Using Python

To transfer data from a vManage web server using a utility such as Python, follow this procedure:

1. Establish a session to the vManage web server.
2. Issue the desired API call.

Establish a Session to the vManage Server

When you use a program or Python script to transfer data from a vManage web server or perform operations on the server, you must first establish an HTTPS session to the server. To do this, you send a call to log in to the server with the following parameters:

- **URL to send the request to:** Use **https://{***vmanage-ip-address***}/j_security_check**, which performs the login operation and security check on the vManage web server at the specified IP address.

- **Request method:** Specify a POST request.
- **API call payload:** The payload contains the username and password in the format j_*username*=*username* and j_*password*=*password*.

Example 4-1 shows the Python code for authenticating to the vManage web server. This example assumes that the vmanage_ip, username, and password variables are already defined with the correct information. Notice the use of the j_security_check option that is added to the login_url variable. The **session** method of the requests library is then used to open a new session on the vManage web server. The POST method of the session object is used to send the request to the login_url, with the payload login_data, which contains the username and password.

Example 4-1 Cisco SD-WAN Python Authentication Code

```
def login(vmanage_ip, username, password):
        """Login to vmanage"""
        base_url_str = 'https://%s:8443/'%vmanage_ip

        login_action = '/j_security_check'

        #Format data for loginForm
        login_data = {'j_username' : username, 'j_password' : password}

        #Url for posting login data
        login_url = base_url_str + login_action

        sess = requests.session()
        #If the vmanage has a certificate signed by a trusted authority change
  verify to True
        login_response = sess.post(url=login_url, data=login_data, verify=False)

        if b'<html>' in login_response.content:
            print ("Login Failed")
            sys.exit(0)
```

API Calls and Responses

When you are logged in to the vManage web server, it is possible to make a GET request in Python to pull data from the vManage REST API.

Example 4-2 shows a function called **get_request()** that receives two parameters as input: the vManage IP address and the resource from which you are trying to obtain the data (in this case, mount_point). Example 4-2 shows how the URL to which the GET request will be issued is built. As you saw earlier, REST API resources can be found after **https://{{*vmanage_ip*}}:{{*port*}}/dataservice/**. You use the **request** method to perform a GET on the url variable. The results of the request are restored in a variable named response and returned to the user. A similar process can be used for POST requests.

Example 4-2 Cisco SD-WAN Python GET Code

```
def get_request(vmanage_ip, mount_point):
    """GET request"""
    url = "https://%s:8443/dataservice/%s"%(vmanage_ip, mount_point)

    response = requests.request("GET", url, verify=False)
    data = response.content
    return data
```

In this case, the value ‘device’ is assumed to be passed to the **get_request** function to complete the URL information. The following command can be used to assign a value to the mount_point variable:

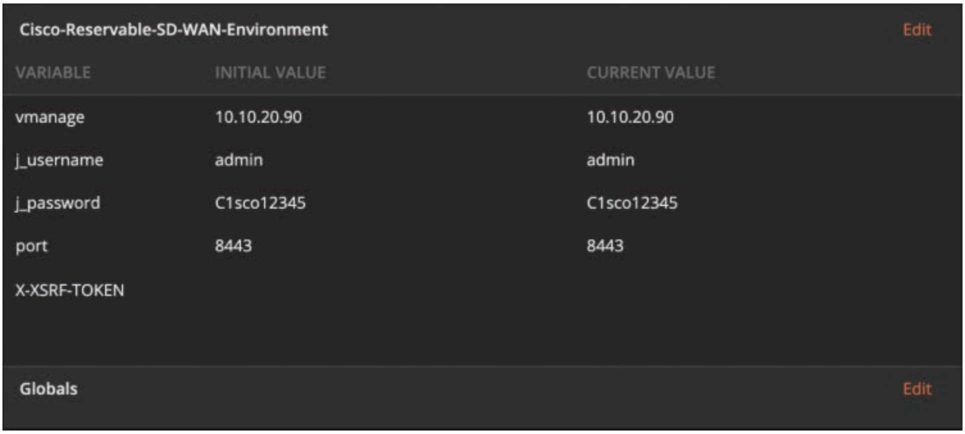
```
response = obj.get_request('device')
```

In order to get information for only a single device, you can filter the results of this call by adding query string parameters. For example, to limit the previous GET response to a single device, you can specify a system IP address. If you change the URL to **https://***vmanage-ip-address***/dataservice/device?system-ip=10.10.1.13**, the JSON object returned is for only that device, so the output is much shorter.

Performing REST API Operations on a vManage Web Server Using Postman

You can use the Postman development tool to make Cisco SD-WAN REST API calls to authenticate, get a list of all the devices that are part of the fabric, get device status, and get interface statistics for all the devices in the fabric. As you saw with Python, the first step in interacting with an API is usually authentication. To simplify this process, it is possible to use a *Postman environment*, which is a list of variables that can be used to easily switch between different environments. By simply modifying the vManage username, password, port, and IP address of your environment, you can access and interact with different Cisco SD-WAN fabrics. The variables defined in the environment can be reused throughout the API calls that are defined in the collection. Figure 4-9 shows the environment variables that are used for the following examples.

Figure 4-9 Postman Cisco SD-WAN Environment Variables for the DevNet Sandbox



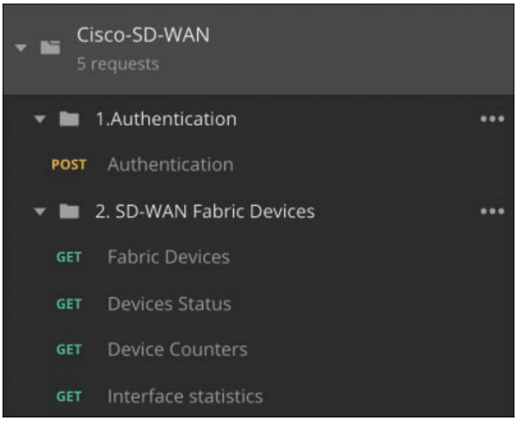
Cisco-Reservable-SD-WAN-Environment

VARIABLE	INITIAL VALUE	CURRENT VALUE
vmanage	10.10.20.90	10.10.20.90
j_username	admin	admin
j_password	C1sco12345	C1sco12345
port	8443	8443
X-XSRF-TOKEN		

Globals

It is also possible to download a *Postman collection*, which is a group of API calls that define endpoints or resources that are available for a specific API. A collection also includes other parameters, headers, or authentication methods that are needed to successfully complete a call. Figure 4-10 shows a collection added to Postman that includes a POST Authentication API and several GET APIs. These are available from the GitHub, at https://github.com/CiscoDevNet/Getting-started-with-Cisco-SD-WAN-REST-APIs.

Figure 4-10 Postman Cisco SD-WAN Collection

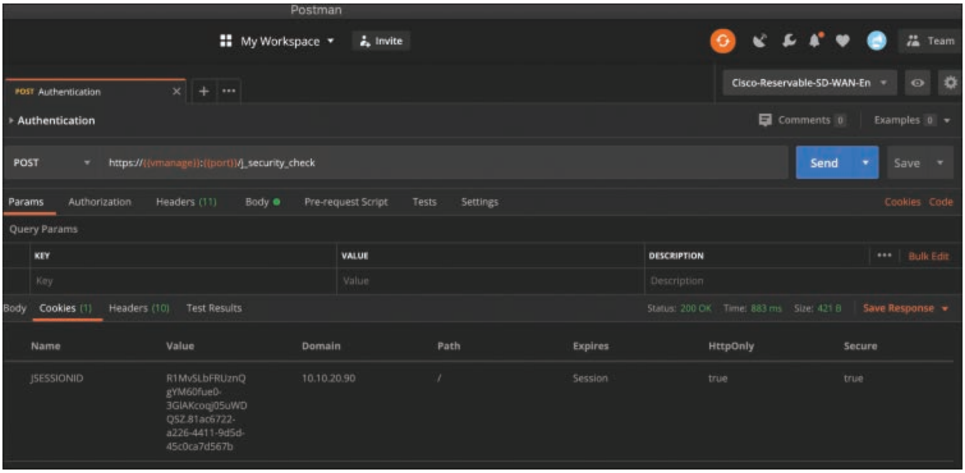


Cisco-SD-WAN
5 requests

- 1.Authentication
  - POST Authentication
- 2. SD-WAN Fabric Devices
  - GET Fabric Devices
  - GET Devices Status
  - GET Device Counters
  - GET Interface statistics

The first API call listed in the collection is called Authentication, and it is in a folder named Authentication. The Authentication call is a POST call. Figure 4-11 shows that to define the endpoint for the authentication call, you use the environment variables {{*vmanage*}} and {{*port*}}, which are replaced with the values defined in the environment: 10.10.20.90 and 8443. The resource that the API call is being sent to is j_security_check. This process is identical to what you saw earlier with the Python authentication script.

Figure 4-11 Postman Cisco SD-WAN Authentication API Call


Postman
My Workspace
Invite
Team
POST Authentication
Cisco-Reservable-SD-WAN-En
Authentication
Comments 0
Examples 0
Send
Save
Params
Authorization
Headers (11)
Body
Pre-request Script
Tests
Settings
Cookies
Code
Query Params
Bulk Edit
Body
Cookies (1)
Headers (10)
Test Results
Save Response

POST https://{{vmanage}}:{{port}}/j_security_check

KEY	VALUE	DESCRIPTION
Key	Value	Description

Status: 200 OK Time: 883 ms Size: 421 B

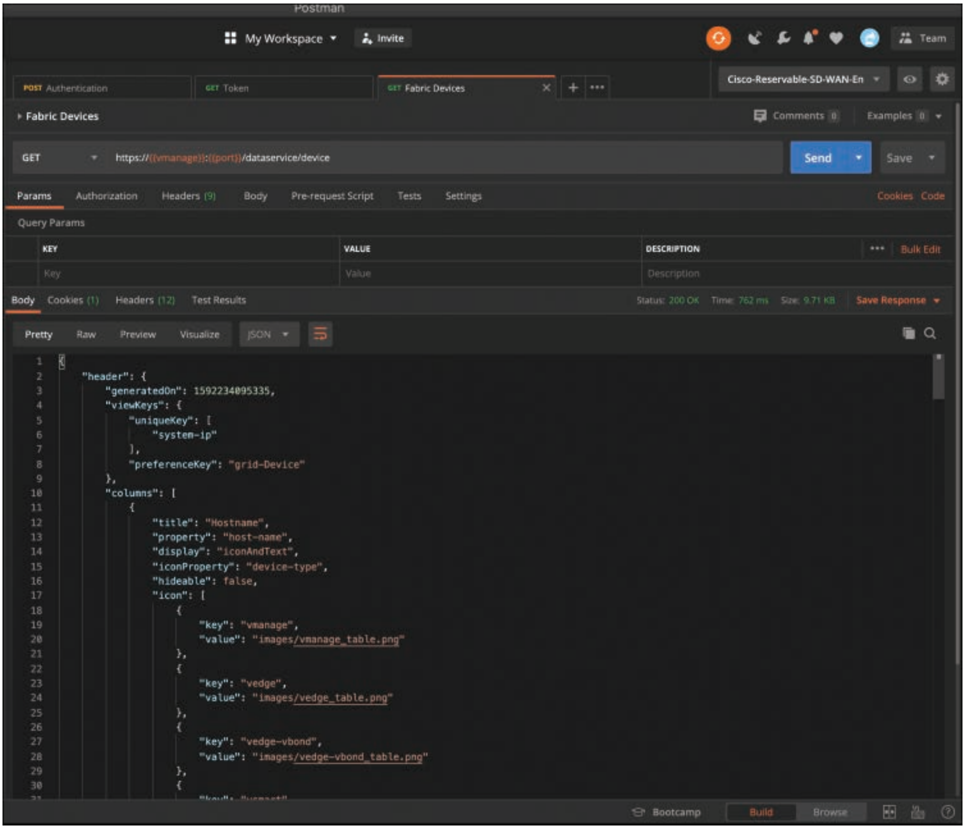
Name	Value	Domain	Path	Expires	HttpOnly	Secure
JSESSIONID	R1MvSLbFRUznQgYM60fue0-3GIAKcoqj05uWDQSZ.81ac6722-a226-4411-9d5d-45c0ca7d567b	10.10.20.90	/	Session	true	true

The username and password are defined on the Headers tab. The username and password are URL encoded in the body of the request and are sent as key/value pairs *j_username* and *j_password*.

If you click the Send button, you should get a status 200 OK message, indicating that the authentication was successful.

Once you are authenticated, it should be possible to use one of the GET APIs from the collection. Figure 4-12 shows that the Fabric Devices GET API was issued, and the vManage web server has responded with a complete list of devices in the sandbox. Again, notice the status 200 OK message.

Figure 4-12 Postman Cisco SD-WAN GET API


Postman
My Workspace
Invite
Team
POST Authentication
GET Token
GET Fabric Devices
Cisco-Reservable-SD-WAN-En
Fabric Devices
Comments 0
Examples 0
Send
Save
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
Code
Query Params
KEY
VALUE
DESCRIPTION
Bulk Edit
Key
Value
Description
Body
Cookies (1)
Headers (12)
Test Results
Save Response
Pretty
Raw
Preview
Visualize
JSON
Bootcamp
Build
Browse

GET https://{{vmanage}}:{{port}}/dataservice/device

Status: 200 OK Time: 762 ms Size: 9.71 KB

```
{
    "header": {
        "generatedOn": 1592234095335,
        "viewKeys": {
            "uniqueKey": [
                "system-ip"
            ],
            "preferenceKey": "grid-Device"
        },
        "columns": [
            {
                "title": "Hostname",
                "property": "host-name",
                "display": "iconAndText",
                "iconProperty": "device-type",
                "hideable": false,
                "icon": [
                    {
                        "key": "vmanage",
                        "value": "images/vmanage_table.png"
                    },
                    {
                        "key": "vedge",
                        "value": "images/vedge_table.png"
                    },
                    {
                        "key": "vedge-vbond",
                        "value": "images/vedge-vbond_table.png"
                    },
                    {
```

Study Resources

For today's exam topics, refer to the following resource for more study.

Resource	Module, Chapter, or Link
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	28

Day 3

Network Automation

ENCOR 350-401 Exam Topics

Automation

- Construct EEM applet to automate configuration, troubleshooting, or data collection
- Compare agent vs. agentless orchestration tools, such as Chef, Puppet, Ansible, and SaltStack

Key Topics

Today we review network configuration management tools such as Puppet, Chef, Ansible, and SaltStack. As you saw yesterday, scripting can help you gather data or modify configurations. In addition, while you can use a language such as Python to perform manual programming, there are also tools you can use specifically for configuration automation and network management.

Today we also look at the Cisco IOS Embedded Event Manager (EEM), which is a unique subsystem in Cisco IOS Software. EEM is a powerful and flexible tool for automating tasks and customizing the behavior of Cisco IOS Software and the operation of a device.

Configuration Management Tools

A configuration management tool may require an *agent*, which is a piece of software that must be installed on the system or device you want to manage. In a network automation use case, this requirement could be problematic because some network devices cannot support running agents and loading general software. In a situation where the network devices do not support an agent, you can use a tool that supports proxy agents or a tool that does not require agents at all.

There are two core types of configuration management tools, and there is also a third type that is usually a derivation of an agent-based tool:

- **Agent-based configuration:** With agent-based tools, an agent must be installed on every device that the configuration management tool will manage.
- **Agentless configuration:** Agentless tools do not require that an agent be installed on every device; instead, they communicate via SSH or another API that a device supports.
- **Proxy-agent configuration:** This type of configuration does not require an agent on every device, but it does require some type of "process" or "worker" to communicate with the master server and the remote device.

Configuration management tools define target end states and allow maximum flexibility to automate 1 device or 10,000 devices. For example, you might want to ensure that your networking devices have the latest version of their software. When you describe what you want to automate in a configuration management tool, you often use a domain-specific language (DSL) or a structure markup language such as YAML. A DSL is a language that is designed specifically to express solutions to problems in a particular domain; in other words, DSLs are special-purpose computer languages and limited compared to a language such as Python or Ruby, which are general-purpose languages.

Some of the benefits of using configuration management tools are fast implementation, lower failure rates, shortened times between fixes, and, importantly, faster user adoption for nonprogrammers. Using these tools also brings the networking environment close to the concept of continuous delivery. It enables infrastructure as code (IaC), which is the idea of writing software for the infrastructure so you can deploy the entire environment by leveraging code rather than manually defining all parameters, which makes it a programmable infrastructure.

From a networking perspective, it is common to deploy changes manually. As an example of changes, you might add a VLAN across a data center or campus, or you might make daily changes to firewall policies for new applications that are deployed. When there is a defined manual workflow to perform a set of tasks, proper tools should be used to automate it. It does not make sense to spend an hour performing a change when it could take just a few minutes to make the change by using a properly engineered tool. Tools such as Puppet, Chef, Ansible, and SaltStack can dramatically reduce the number of manual interactions in a network.

Configuration management tools are often referred to as *DevOps tools*. Organizations that have implemented some form of DevOps practices use these configuration management and automation tools to automate application, infrastructure, and network deployments to a high degree without the need to do manual programming using a language such as Python. These tools reduce the time it takes to perform certain tasks and offer greater predictability.

Configuration Management for Networking

Puppet, which was created in 2005, has been in use longer than Chef and Ansible. Puppet manages systems in a declarative manner, which means you define an ideal state of the target system without worrying about how it happens. (In reality, this is the approach that all these tools take.) Puppet is written in Ruby and refers to its automation instruction set as Puppet *manifests*. The major point to realize is that Puppet is agent based.

As mentioned earlier, *agent based* means that a software agent must be installed on all devices that you want to manage with Puppet, including servers, routers, switches, and firewalls. It is often impossible to load an agent on every network device, so the number of devices that can be used with a new Puppet package is limited. This can be remedied thanks to the use of Puppet proxy devices. Regardless, this means that Puppet has a greater barrier to entry when getting started.

Chef, another popular configuration management tool, follows much the same model as Puppet. Chef is based in Ruby, uses a declarative model, is agent based, and refers to the Chef automation instruction as *recipes* (groups of which are called *cookbooks*).

It is often difficult to load agents onto all machines to automate them. However, even when it is technically possible to do so, deploying the solution or tool can be time-consuming. Ansible was created as an agentless alternative to Puppet and Chef. Red Hat acquired Ansible in 2015, and IBM more recently acquired Red Hat. Being natively agentless significantly lowers the barrier to entry from an automation perspective. Because Ansible is agentless, it can integrate and automate a device by using an API. For example, integrations can use REST APIs, Network Configuration Protocol (NETCONF), SSH, or even Simple Network Management Protocol (SNMP), if desired. A *playbook* is a set of Ansible *tasks* (instructions) that is used to automate devices. Each playbook consists of one or more *plays*, each of which consists of individual *tasks*. Ansible is written in Python.

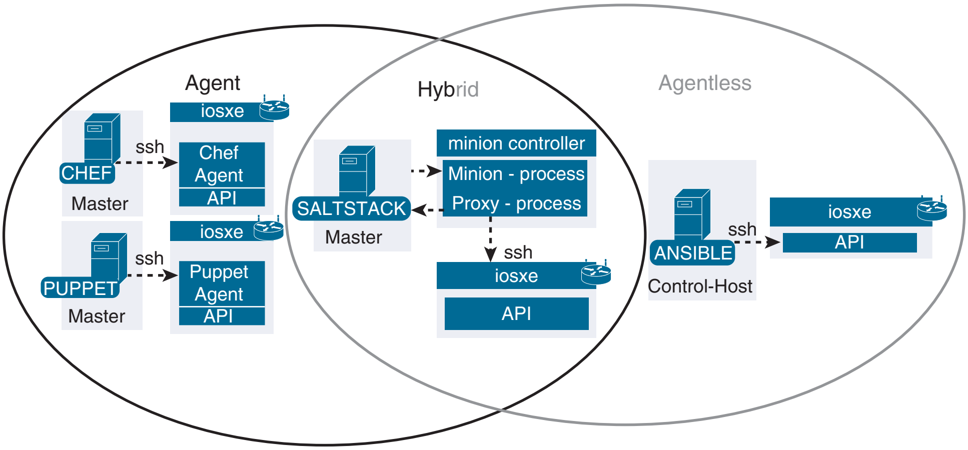
It is worth noting that each of these tools had its start in automation for applications and cloud infrastructure. It was not until each of them had a significant amount of traction in the market that they started to include network automation in their portfolios.

Salt, by SaltStack, is an agent-based configuration management and orchestration tool that is similar to Puppet and Chef. Each Salt-managed node must have a Salt *minion* service installed. Usually, this approach is not an issue on a server, but in the networking world, it is not always possible to install custom software on the network equipment that you want to manage. To solve this problem with Salt, you can use a *proxy minion*, which is basically a minion that does not need to be installed on the targeted device and can run ubiquitously. In other words, proxy minions run intermediate software that mediates between the Salt master and the Salt minion. The Salt master is a master daemon that is used for sending configurations and commands to the Salt minions. The Salt master is basically a machine that manages the infrastructure and dictates policies. Salt minions receive the configuration and commands from the master daemon. It is the responsibility of Salt minions to execute the instructions sent by the Salt master, report job success, and provide data related to the underlying host.

Salt is written in Python, and it uses a message queue–based system called ZeroMQ to push commands and configuration to minions. It also allows for publish and subscribe messaging.

Figure 3-1 compares these three configuration management tools. Notice the relationships of the different deployment models: agent based, agentless, and hybrid.

Figure 3-1 Configuration Management Tools Comparison



Next we look at each of these tools in more detail.

System Management with Ansible

Ansible is open-source provisioning software that allows for centralized configuration management. Ansible was originally written by Michael DeHann of AnsibleWorks, which was acquired by Red Hat in 2015. Red Hat in turn was acquired by IBM in 2019. Ansible is free and open source and is included as part of Fedora. It is also available for RHEL, CentOS, Scientific Linux, and other operating systems through their respective package management systems or the Python package management tool **pip**.

Unlike other management platforms and services, Ansible does not require an agent to be installed on the system that it manages, nor does Ansible need or use a centralized controller for configuration management. It is possible to perform automation from any management system by referencing inventories, modules, or playbooks.

Its declarative approach means that you only tell Ansible what you want to achieve as a final goal instead of encoding all instructions to reach it. For example, you do not have to tell Ansible where a specific service resides, how to start it, and what to do after it starts. You simply say, "I want this service to be started, and then I want another service to be restarted."

System Management with Ansible: Components

The components of Ansible come together to make a very powerful management system. Understanding each component and the relationships between the components is essential to realizing their power:

- **Ansible control station** is the management station and launching point for all Ansible activities. Unlike many other management platforms, Ansible does not require a dedicated server or elaborate hardware to manage an environment. You could literally manage an enterprise from your personal laptop.
- An Ansible *module* is a standalone collection of commands that is written in a standard scripting language (for example, Python) and used to execute a state change. An administrator can write an Ansible module using any language that supports the JavaScript Object Notation (JSON) data format.
- *Playbooks* are files that are used to define the desired or final state and also used to orchestrate operation across multiple nodes.
- *Inventory files* contain systems managed by Ansible. An administrator uses an inventory file to group managed systems. Inventory files can be dynamically built by contacting a remote API that responds with a valid JSON response.
- *YAML files* are commonly referred to as configuration files and destinations for data being stored. Ultimately, YAML is a data format that is transported over Secure Shell (SSH) by default and with PowerShell support for Windows nodes over the WS-Management protocol.
- *Ansible Tower* is a web service console (GUI) that follows the REST standard for programmability. It is a licensed product from Red Hat, based on the open-source AWX Project.

System Management with Ansible: Tools

Managing Ansible is a simple process that basically involves selecting a terminal program to connect to your management server running Ansible. You can select your preferred text editor for creating and editing inventory files, playbooks, and modules, and you can use your preferred version control service to manage access to code and control collaboration and editing. Examples of these tools include PuTTY for SSH access, Visual Studio Code for configuration file management, and GitLab for storage, collaboration, and version control. All this can be done from any Linux-based machine that has Python and Ansible installed.

How Ansible Works

Working with Ansible requires only a few quick installation and update steps. Once the installation and update are complete, you can configure Ansible operations and defaults. The inventory file and modules work together to help you execute your changes to the specified target systems.

You can install Ansible in multiple ways, the easiest of which is to use Python **pip**. With a Python virtual environment active, execute **pip install ansible**, and you are done. You can then execute **ansible -version** to verify that the installation was successful. Many operating systems offer Ansible prepackaged in their repositories (for example, **yum install ansible**, **apt-get install ansible**). Ansible configuration files are used to configure Ansible operations. The base configuration file is located at /etc/ansible/ansible.cfg.

Inventory files enable an administrator to define an inventory of systems against which Ansible modules will be executed. The basic contents of an inventory file are hosts and groups. Host entries point to the Domain Name System (DNS) name or IP address of a managed end system, and a group is a collection of hosts under a collective label.

How Ansible Works: Push Model

With Ansible installed and upgraded and a list of devices in your network defined in an inventory file, you can manage configuration modules and execute them to make changes to your managed devices.

An Ansible module can be thought of as a small program pushed to and run on a managed device to achieve the desired configuration state for that device. Most modules are standalone. Ansible also gives an administrator the ability to write modules by using standard scripting languages such as Python. With more than 750 built-in modules, organized by vendor and technology, Ansible enables administrators to ramp up and manage their environments quickly and easily.

You can use the **ansible-doc** *modulename* command to quickly view information about how to use an installed module. For more information, see https://docs.ansible.com.

When you have modules installed, you can execute them from the Ansible host against the systems defined with your inventory file by using the **ansible-playbook** *example-playbook.yml* command. Ansible connects to all systems defined in the inventory simultaneously, makes the prescribed changes, and displays the status PLAY RECAP on the terminal screen.

Ansible also supports ad hoc execution of modules. To use this feature, run the command **ansible** *host1*,*group2* **-m** *modulename* **-a** *moduleargs*, where *host1* and *group2* are entries defined in your inventory file, *modulename* is the name of the Ansible module you want to execute, and *moduleargs* are the required arguments for the specific module.

Some modules don't require arguments and can be called only with *modulename*; for example, **ansible** *host1*,*host2*,*host3* **-m ping** returns success if all hosts are reachable. In this example, **ping** is the name of the module being called.

Ansible Playbooks: Terms

Ansible uses some specific terminology for orchestration through playbooks, as shown in Table 3-1.

Table 3-1 Ansible Terms

Term	Description
Module	Code, written using any of a number of scripting languages (for example, Python), to perform an action on a managed device.
Task	An action referencing a module to run together with input argument and actions.
Play	A set of tasks for a host or group of hosts.
Playbook	A file, written in YAML, that includes one or more plays.
Role	A set of playbooks, often prebuilt, used to execute a standard configuration in a repeatable manner. You can assign multiple roles to a single host.

For example, an inventory file might contain server1, server2, and server3 host entries that are part of the internet_servers group, which should have ssh_server, http_server, and php_server roles applied.

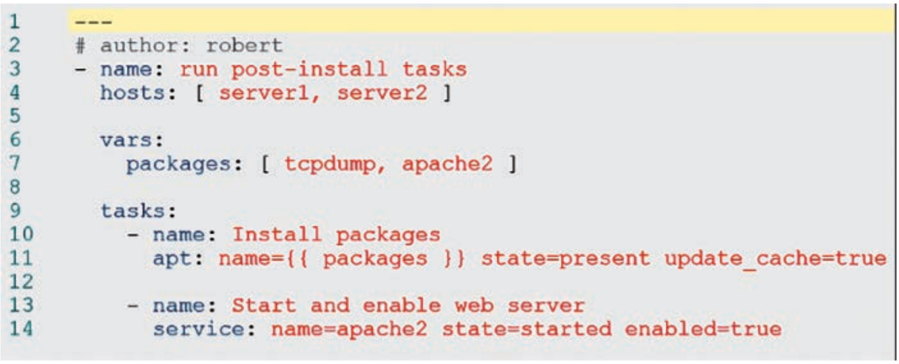
Ansible Playbooks: Components

An Ansible playbook is a simple human-readable text file in YAML format, with keywords that Ansible recognizes as instructions. Because YAML is a plaintext file, it is ideal for collaboration and makes troubleshooting much easier because line numbering and version control provide insight into what was changed and when and who made the changes.

A playbook can be named and consists of hosts and tasks that should be performed. A lot of optional keywords can be used to change the way Ansible interacts with hosts defined in an inventory file. For example, **gather_facts** is an instruction to Ansible that enables or disables runtime variables available to tasks. Setting it to **no** makes execution faster, but the tasks lose access to variables that Ansible collects before execution; for example, ansible_distribution would be empty instead of containing the value Ubuntu or RedHat.

Figure 3-2 shows an example of an Ansible playbook.

Figure 3-2 Ansible Playbook Example



```
---
# author: robert
- name: run post-install tasks
  hosts: [ server1, server2 ]

  vars:
    packages: [ tcpdump, apache2 ]

  tasks:
    - name: Install packages
      apt: name={{ packages }} state=present update_cache=true

    - name: Start and enable web server
      service: name=apache2 state=started enabled=true
```

A pound sign (#) indicates a comment; Ansible ignores everything from the # to the end of the line.

The vars section contains user-defined variables that are later referenced in the tasks section. Variable expansion is done by enclosing a variable name inside double curly brackets—for example, {{ *variable* }}.

The tasks section contains tasks that are executed in the order in which they are listed, and they are executed in linear fashion on each host; that is, when one task finishes on all hosts, execution of the next task starts. This behavior can be changed by setting **strategy** to **free** in the top-level section of the playbook.

Ansible can run in parallel on a maximum of five hosts. If you add more hosts, you must wait longer for each task to complete. You can change this behavior by setting **forks** to **20** in the top-level section of the playbook.

Each section under tasks starts with an optional name, followed by a module name that you want to use (**apt** and **service** in Figure 3-2). Module parameters belong under the module name and must be indented to indicate that they belong to the module and not the task. Some module parameters are required; others are optional. You can find out which parameters are required by using an equals (=) sign in the output of **ansible-doc apt**.

The playbook in the example in Figure 3-2 declares the variable **packages** and connects to server1 and server2. The first task updates the apt cache with the latest package information from remote repositories and installs the **tcpdump** and **apache2** packages. The second task starts the **apache2** web server and enables the **apache2** web server to start on boot.

Ansible Playbooks: Inventory File

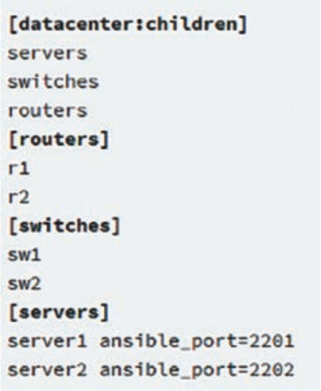
An inventory file is a collection of all the hosts that are managed by Ansible. It is a simple plaintext file in which you specify hosts, logical groupings, and special variables for Ansible.

Figure 3-3 shows an example of an inventory file that contains the following information:

- A group named servers with two hosts, each with a special variable ansible_port to indicate that **sshd** is running on a nonstandard port
- A group named switches with two hosts

- A group named routers with two hosts
- A group named datacenter with groups servers, switches, and routers as members

Figure 3-3 Ansible Inventory Example



```
[datacenter:children]
servers
switches
routers
[routers]
r1
r2
[switches]
sw1
sw2
[servers]
server1 ansible_port=2201
server2 ansible_port=2202
```

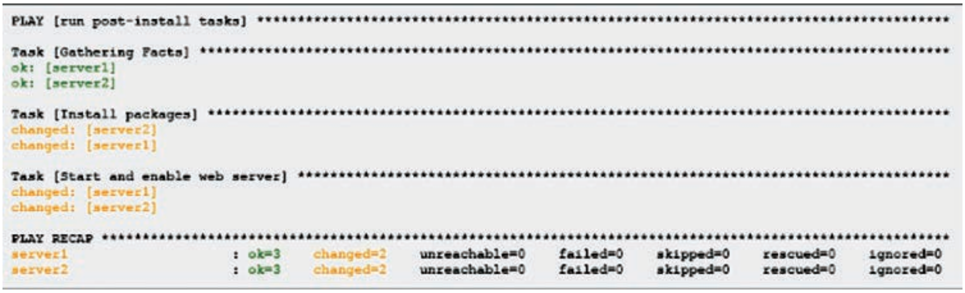
You can target any of the hosts or groups in your inventory by changing the hosts keyword in your playbook. The group **datacenter** is convenient when you want to execute something on all the desired groups or set a variable for all groups in one place. There is also a predefined default group named **all**, which you can use to target all hosts in your inventory.

Ansible: Executing the Playbooks

You can execute a playbook by using the **ansible-playbook** -**u** *username* **root** *example*.**yml** command, where **-u** *username* is your username for administrative login and *example*.**yml** is the name of your playbook. Ansible uses the root username by default, so you can omit **-u** in this command if you want to use root.

Figure 3-4 shows an example of an Ansible Playbook running.

Figure 3-4 Ansible Playbook Results



```
PLAY [run post-install tasks] ***************************************************************************

Task [Gathering Facts] **********************************************************************************
ok: [server1]
ok: [server2]

Task [Install packages] *********************************************************************************
changed: [server2]
changed: [server1]

Task [Start and enable web server] **********************************************************************
changed: [server1]
changed: [server2]

PLAY RECAP **********************************************************************************************
server1                   : ok=3    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
server2                   : ok=3    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

In the example shown in Figure 3-4, Gathering Facts runs before any tasks to collect system information and populate variables that the tasks might need. You can disable this behavior by setting gather_facts to false. The Install packages task runs first, Start and enable web server runs second, and PLAY RECAP tells you how many tasks made changes on each host (two in this case).

System Management with Puppet

Puppet is a configuration management tool that you can use to automate configuration deployment in your infrastructure. The Puppet configuration management tool has the following features:

- Puppet is an open-source project managed by a company called Puppet.
- It is written in C++, Clojure, and Ruby.
- It uses Puppet's declarative language to describe a desired configuration.
- It can be used on many different systems, including various Linus distributions (such as Red Hat and Ubuntu), various UNIX systems, and Windows.

Puppet Architecture

Puppet has a client/server architecture. The client in Puppet is called the *agent*, and the server is called the *master*. The master controls the complete configuration, and the agent periodically pulls the configuration from master. The following workflow is used to update the agent's configuration:

1. An agent node requests a catalog from the master.
2. The master compiles the Puppet code and returns the catalog with the required configuration.
3. The agent applies the configuration by checking the state of the resources and applying the required changes.
4. The agent responds with the report.

This procedure is repeated periodically. By default, an agent requests the configuration every 30 minutes. Typically, you would want to manage the complete infrastructure with one tool. Puppet requires that an agent be installed on each managed node. However, you cannot install an agent on some devices, such as switches, routers, and firewalls. For those devices, you need to configure a node as a proxy for devices. The proxy performs all communication with the master, connects to the device, and applies the configuration.

Basic Puppet Concepts

When you use Puppet, it is important to be familiar with its basic concepts:

- **Resource:** This is the portion of Puppet code that describes a particular aspect of a system.
- **Class:** This is a collection of resources that can be reused multiple times.
- **Manifest:** This configuration file describes how to configure your resources.
- **Catalog:** This is a compiled configuration for each individual node.
- **Module:** This is a collection of manifests, templates, and other files that is used to configure multiple options of a specific feature and can be reused.

The most elementary Puppet entity is called a *resource*. It is a unit of configuration that you use in configuration files. You need to define a resource type, a title, and one or more attributes with specific values. A resource is defined with the following syntax:

```
<TYPE> { '<TITLE>': <ATTRIBUTE> => <VALUE>, }
```

You can list multiple attributes, separated with commas. You can also write attributes on multiple lines. Resources must be unique to avoid having conflicting values for the same resource. The uniqueness is achieved with the resource title and the name attribute. If resources are not unique, you get an error when compiling the configuration.

You can group resources into *classes*. Classes are similar to classes in regular programming languages, as you can use them to better organize the code. Before you can use a class, you need to define a class. Defining a class means specifying the class's configuration and making it available for later use. The class definition does not specify any actual configuration for nodes.

The main configuration of the infrastructure is specified in a *manifest*. The manifest stores the Puppet code that describes how resources should be configured. The manifest is a text file with the .pp extension. It uses UTF-8 encoding. The manifest can be a single file, or you can specify configuration in a directory that contains multiple manifests. Puppet treats multiple files as one. The main manifest is usually called site.pp.

The configuration in manifests is compiled into a *catalog* for each individual node. The catalog describes the desired state of a specific node and is generated from the logic that is defined in the manifest. The master node sends the catalog to the agent. The agent applies the required configuration to the resource to bring the node to the desired state.

When you start writing complex manifests, you probably want to store your code in *modules*. Modules allow reusability. They are used to configure a specific part of the infrastructure. Modules contain all previously described elements, including resources, classes, and manifests. You can write your own modules, but you must follow a defined structure and place the module in the correct location on the system. There are also a lot of modules available from Puppet Forge, written by Puppet developers and members of the open-source community. When you develop or install a module, the module gives you classes and resources that you can use in your configuration code. You can use these classes in the same way you would define classes inline in the main manifest.

Puppet Example

Example 3-1 shows a Puppet configuration with the following objectives applied to two Linux servers called puppeta01 and puppeta02:

- Configure the IP address on the network interface.
- Configure the default gateway.
- Configure the DNS server.
- Configure the NTP address.

This code would be added to a file called site.pp.

Example 3-1 Puppet Configuration Script

```
class conf_if (String $ip_address) {
  file { 'if_config':
    path => '/etc/sysconfig/network-scripts/ifcfg-enp0s3',
    content => template("$settings::manifest/network-if.cfg"),
    ensure => present,
  }
  exec { 'if_down': command => '/usr/sbin/ifdown enp0s3',
    require => File['if_config'],
  }
  exec { 'if_up': command => '/usr/sbin/ifup enp0s3',
    require => Exec['if_down'],
  }
}

class conf_dns {
  $dns_servers = ['192.168.1.253', '192.168.1.254']
  file { 'dns_config':
    path => '/etc/resolv.conf',
    content => template("$settings::manifest/dns.cfg"),
    ensure => present,
  }
}

node 'puppeta01' {
  class { 'conf_if': ip_address => "192.168.1.11" }
  include conf_dns
  class { 'ntp': servers => ['192.168.1.250'] }
}

node 'puppeta02' {
  class { 'conf_if': ip_address => "192.168.1.12" }
  include conf_dns
  class { 'ntp': servers => ['192.168.1.250'] }
}
```

The first class in Example 3-1, **conf_if**, accepts the **ip_address** variable. The first action is to create a file in the correct location, which is specified with the **path** attribute. The **content** attribute specifies the content of the file. The content is created with the **template** function, which evaluates the specified template with the defined variables. In this case, you need to pass the **ip_address** variable. The **content** attribute uses the **$settings::manifest** variable to point to the correct location on the file system. Next, you need to disable and enable the interface to apply the new configuration. The **exec** resource uses the **require** attribute, which adds a dependency. The code first creates a file, then executes the **ifdown** command, and finally executes the **ifup** command.

The second class in the example, **conf_dns**, creates a template that will be used in the /etc/resolv.conf file and applies the 192.168.1.253 and 192.168.1.254 IP addresses to the DNS configuration of the device.

The final two sections define the nodes that will receive the required configuration. Both nodes call the **conf_if** class for IP address assignment, call the **conf_dns** class, and call an external NTP module called **ntp** that was previously installed.

Now that the configuration is ready, the last step is to execute the **puppet agent -t** command on the node to run the pull request and update the configuration; alternatively, you can just wait until the next pull period.

System Management with Chef

Chef is a powerful automation platform that transforms complex infrastructure into code, enabling data center infrastructure automation with a declarative, intent-based model. Whether you're operating in the cloud, on premises, or in a hybrid environment, Chef automates how applications are configured, deployed, and managed across your network, no matter its size.

Chef is built around simple concepts: achieving desired state, using centralized modeling of IT infrastructure, and using resource primitives as building blocks. These concepts enable you to quickly manage any infrastructure with Chef. These very same concepts allow Chef to handle the most difficult infrastructure challenges and customer use cases; anything that can run the Chef client can be managed by Chef.

Chef, like Ansible and Puppet, exists as software packages that you install and run. Chef (the company) offers several products; Chef Automate is the product that most people refer to simply as Chef. As with Puppet, in production, you are likely to run Chef as a server (in what is called server-client mode), and engineering staff use multiple Chef workstations to build Chef files that are stored on the Chef server. However, you can also run Chef in standalone mode (called Chef Zero), which is helpful when you're just getting started and learning in the lab.

Chef uses an architecture similar to that of Puppet. For network devices, each managed device (called a Chef node or Chef client) runs an agent. The agent performs configuration monitoring in that the client pulls recipes and resources from the Chef server and then adjusts its configuration to stay in sync with the details in those recipes and run lists. Note, however, that Chef requires on-device Chef client code, and many Cisco devices do not support a Chef client, so you are likely to see more use of Ansible and Puppet for Cisco device configuration management.

Chef Concepts

When you use Chef, it is important to be familiar with its basic concepts:

- **Chef server:** The Chef server acts as a hub for configuration data. It stores the following:
    - Cookbooks
    - Recipes (the policies that are applied to nodes)
    - Metadata that describes each registered node that is being managed by the Chef client

- **Node:** A node is any physical, virtual, or cloud machine or switch configured to be maintained by a Chef client.
- **Chef client:** The Chef client runs locally on every node that is registered with the Chef server. It performs all configuration tasks specified by the run list and brings the client into the desired state.
- **Chef resources:** This is a grouping of managed objects/attributes and one or more corresponding implementations. It describes the desired state for a configuration item and declares the steps needed to bring that item to the desired state. It specifies a resource type, such as a package, template, or service, and lists additional details (also known as attributes) as necessary. Resources are grouped into recipes, which describe working configurations.
- **Cookbook:** A cookbook defines a scenario and contains everything that is required to support that scenario. A cookbook is used for device configuration and policy distribution and includes the following elements:
  - Recipes that specify the resources to use and the order in which they are to be applied
  - Attribute values
  - File distributions
  - Templates
  - Extensions to Chef, such as libraries, definitions, and custom resources
- **Recipe:** A recipe is a collection of resources, defined using patterns (resource names, attribute/value pairs, and actions). Helper code is added to a recipe by using Ruby. Recipes must follow the following guidelines:
  - A recipe must be stored in a cookbook.
  - A recipe may use the results of a search query and read the contents of a data bag.
  - A recipe may have a dependency on one (or more) recipes.
  - A recipe must be added to a run list before it can be used by the Chef client.
  - A recipe is always executed in the same order as listed in a run list.
  - The Chef client runs a recipe only when asked to do so.

Chef Example

Example 3-2 shows a sample cookbook configuration of a switch's Layer 3 and Layer 2 interfaces. The Ethernet 1/1 interface is configured as a Layer 3 interface with an IP address and a subnet mask. Notice that Layer 3 functionality is enabled thanks to the **switchport_mode 'disabled'** command. Interface Ethernet 1/2 is assigned to VLAN 100 and is configured as an access port.

Example 3-2 Chef Configuration Script

```
cisco_interface 'Ethernet1/1' do
    action :create
    ipv4_address '10.1.1.1'
    ipv4_netmask_length 24
    ipv4_proxy_arp true
    ipv4_redirects true
    shutdown true
    switchport_mode 'disabled'
  end

  cisco_interface 'Ethernet1/2' do
    action :create
    access_vlan 100
    shutdown false
    switchport_mode 'access'
    switchport_vtp true
  end
```

System Management with SaltStack

Created by Thomas Hatch in 2011, Salt, now known as SaltStack, is a modular, Python-based configuration management tool that is designed for high-speed data collection/execution. The tool has gained considerable traction in the enterprise because of its performance benefits over competing solutions, including Ansible.

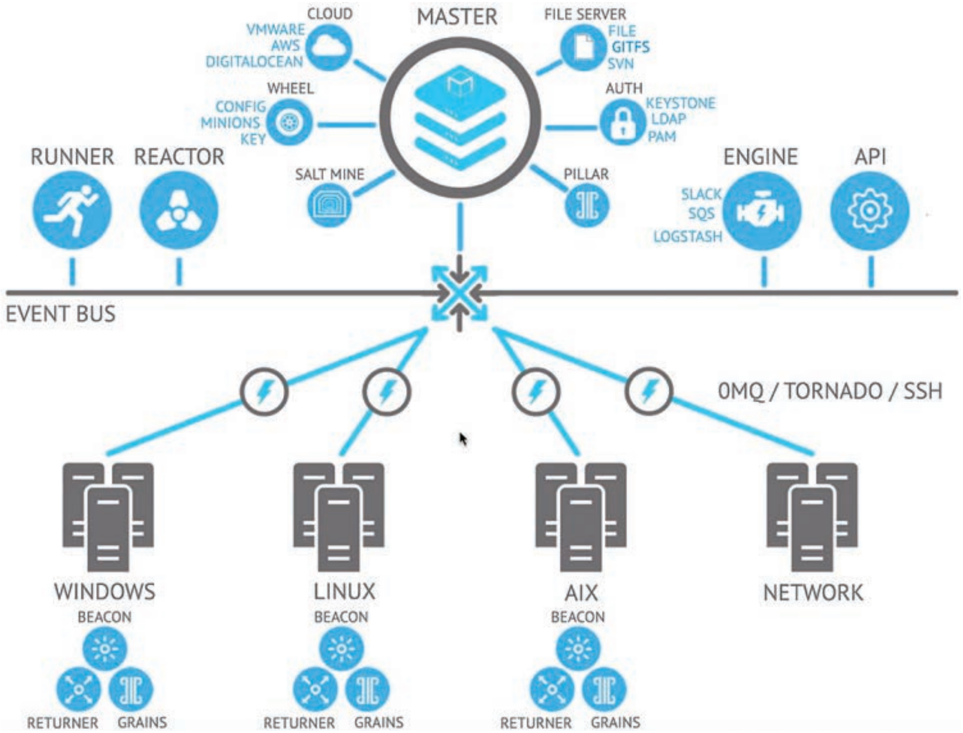
SaltStack's speed and performance benefits are made possible by its lightweight ZeroMQ messaging library, which is a framework for establishing concurrent, persistent TCP connections between the server and agents (that is, the Salt master and minions). The platform is available as an open-source project or as an enterprise commercial offering known as SaltStack Enterprise.

Salt Architecture

At the core of the Salt ecosystem are two critical components: the master and the minion. The other components play supporting roles to make the ecosystem work. Figure 3-5 provides a high-level view of a Salt architecture example, which includes the following components:

- **Master:** The master is the server that runs the core services to communicate with the minions. It also contains the key store for encryption between the minions.
- **Minion:** A minion is a Python-based endpoint that can be deployed on all major Linux and Windows operating systems. The minions are connected to the master, and you can have the master ask them to do a variety of tasks by adding an easy-to-understand YAML file or by directly running commands from the master.
- **SSH (agentless):** Salt supports an agentless mode over SSH for assets that don't have minions installed. Salt SSH supports the same system normalization functions that are supported by the minions.

Figure 3-5 SaltStack Architecture


CLOUD
VMWARE
AWS
DIGITALOCEAN
MASTER
FILE SERVER
FILE
GITFS
SVN
WHEEL
CONFIG
MINIONS
KEY
AUTH
KEYSTONE
LDAP
PAM
RUNNER
REACTOR
SALT MINE
PILLAR
ENGINE
SLACK
SQS
LOGSTASH
API
EVENT BUS
0MQ / TORNADO / SSH
WINDOWS
BEACON
RETURNER
GRAINS
LINUX
BEACON
RETURNER
GRAINS
AIX
BEACON
RETURNER
GRAINS
NETWORK

- **Proxy minion:** On assets where agents cannot be deployed or where it is not possible to use SSH for remote access, a Salt proxy minion can be used. The proxy minion is designed to connect to network devices and IoT devices.
- **Monitor (beacons):** Minions can be configured to monitor files, processes, services, and a host of other things. They can also generate events when certain criteria are met, such as failed logins, unauthorized changes to critical files or processes, or even unexpected changes in CPU load or disk usage.
- **Respond (reactor):** Another feature of Salt is its ability to react to events and take actions. The reactor can be configured to take predefined actions based on certain event criteria (such as restarting the MySQL service if it stops). This helps organizations achieve a state of event-driven SecOps IT automation.
- **States or formulas:** These are files that contain YAML and templated data to configure minions.
- **Grains:** Salt comes with an interface to derive information about the underlying system. This is called the grains interface because it presents Salt with grains of information. Grains are collected for the operating system, domain name, IP address, kernel, OS type, memory, and many other system properties. The grains interface is made available to Salt modules and components so that the right Salt minion commands are automatically available on the right systems.

- **Pillars:** A pillar is an interface for Salt that is designed to offer global values that can be distributed to minions. A pillar is a free-form resource of data (that can be JSON, YAML, or whatever you need), and it can either be stored in a file or externally. A pillar allows integration with other systems where a shared data store would be of value (for example, an ITSM or asset register).
- **Mine:** This is used for data fetching so that you can also return data from minions and store it to be used in other tasks, such as for template-based state configuration.

SaltStack Example

Example 3-3 shows a sample configuration that sets the NTP server IP addresses on specific devices. The example creates the necessary pillar files, **ntp_server.sls** and **top.sls. ntp_server.sls** contains a list of IP addresses that the devices should use to synchronize their clocks. The **top.sls** file lists the devices that need to be configured with the NTP addresses. The * tells Salt to provide the content from **ntp_servers.sls** to all minions.

Example 3-3 SaltStack Configuration Script

```
ntp.servers:
  - 172.17.17.1
  - 172.17.17.2
```

```
base:
  '*':
    - ntp_servers
  'device1':
    - device1_pillar
  'device2':
   - device2_pillar
  'device3':
    - device3_pillar
```

```
$ sudo salt 'device*' saltutil.refresh_pillar
```

```
$ sudo salt 'device*' pillar.get ntp.servers
device1:
    - 172.17.17.1
    - 172.17.17.2
device2:
    - 172.17.17.1
    - 172.17.17.2
device3:
    - 172.17.17.1
    - 172.17.17.2
```

The first **salt** command in this example refreshes the local information on the minions, and the second **salt** command checks to make sure the data has been loaded correctly on the devices. The output confirms that the three devices are now configured to contact the two NTP servers at 172.17.17.1 and 172.17.17.2.

Embedded Events Manager

Cisco IOS Embedded Event Manager (EEM) supports more than 20 event detectors that are highly integrated with different Cisco IOS Software components to trigger actions in response to network events. These scripts are programmed using either a simple command-line interface (CLI) or a scripting language called Tool Command Language (Tcl). In this section, we first review EEM and then look more closely at Tcl scripting.

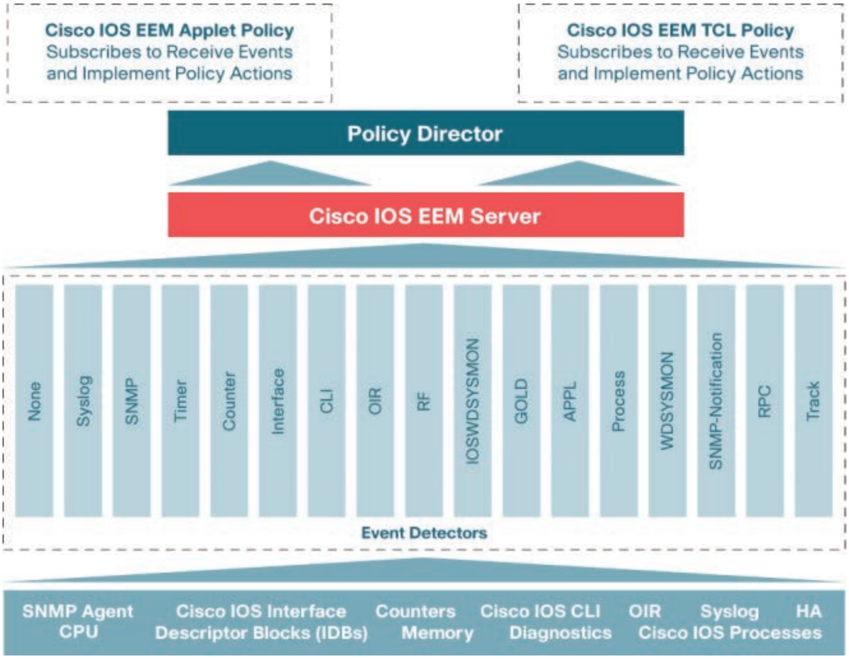
EEM Architecture

EEM consists of three components:

- **Policies:** Policies, or scripts, are either applets or Tcl scripts configured by the administrator.
- **EEM server:** The EEM server is the director of EEM. When a configured event occurs, the associated action is implemented.
- **Event detectors:** An event detector is used as a trigger, based on certain conditions. Some of these conditions include monitoring for syslog events, online insertion and removal (IOR), CLI input, and timers.

Figure 3-6 illustrates the relationship between these three components.

Figure 3-6 EEM Architecture


Cisco IOS EEM Applet Policy
Subscribes to Receive Events
and Implement Policy Actions
Cisco IOS EEM TCL Policy
Subscribes to Receive Events
and Implement Policy Actions
Policy Director
Cisco IOS EEM Server
None
Syslog
SNMP
Timer
Counter
Interface
CLI
OIR
RF
IOSWDSYSMON
GOLD
APPL
Process
WDSYSMON
SNMP-Notification
RPC
Track
Event Detectors
SNMP Agent
CPU
Cisco IOS Interface
Descriptor Blocks (IDBs)
Counters
Memory
Cisco IOS CLI
Diagnostics
OIR
Syslog
HA
Cisco IOS Processes

Policies

Policies, or scripts, are either applets or Tcl scripts configured by the administrator:

- **Applets:** With applets, users do not need to know how to write Tcl scripts. The body of an applet appears in the configuration of a Cisco IOS device. Applets were designed to provide a simpler interface for EEM.
- **Tcl scripts:** Policies can be defined through the IOS CLI. Scripts provide more control than do applets.

EEM Server

The EEM server is a bridge between the policies and internal Cisco IOS subsystems used in the event detectors. Some of the functions of the EEM server are to register events seen in the IOS subsystems, store information about an events, publish events, request additional information about events, register internal script directories, register Tcl scripts and applets, and process actions taken by user-defined scripts.

Event Detectors

Event detectors monitor the operational state of processes on an IOS device. These processes run at medium priority and are always operational. On detecting an event, an event detector sends an alert that provides information about the specific event. The list that follows describes a few of the event detectors in Cisco IOS:

- **Syslog event detector:** This event detector is triggered when a syslog message match is seen in the output of the console. Additional granularity can be achieved depending on the number of occurrences for any given message. Console logging does not have to be active for the syslog event detector to generate events; this is completely independent of which logging targets are configured.
- **SNMP event detector:** A specific SNMP variable can be polled, and if the SNMP variable reaches a threshold, a trigger can be generated.
- **IP SLA event detector:** This detector publishes an event when an IP service-level agreement (SLA) reaction is triggered.
- **Interface event detector:** This detector monitors interface counters based on a defined threshold value. The threshold can be specified as an absolute value or as an incremental value.
- **CLI event detector:** This event detector triggers when a specified command is entered via a console or Telnet/SSH session. This detector screens CLI commands that match a regular expression. When a match is found, an event is published, and the CLI event detector can perform a logical match before the command is parsed and executed.
- **NetFlow event detector:** When a NetFlow event is triggered, the NetFlow event detector publishes an event.
- **Enhanced object tracking event detector:** This event detector is triggered when the status of the tracked object changes. A unique number identifies each tracked object. This number

is specified on the tracking CLI, and the detector process uses this number to track a specific object. The tracking process periodically polls the tracked objects and notes any change in value. The changes in the tracked object are communicated to the event detector process either immediately or after a specified delay. The object values are reported as either up or down. Enhanced object tracking is now integrated with EEM to allow EEM to report on the status change of a tracked object. For example, the Hot Standby Router Protocol (HSRP) active router may change because of the loss of a neighbor router or due to an interface state change.

- **Routing event detector:** This event detector is invoked when information in the Routing Information Base (RIB) has been changed.

The EEM event detectors have evolved based on IOS releases. The functionality of event detection and action depends on the specific release. Use the Cisco Feature Navigator (https://cfnng.cisco.com/) or the **show event manager version** command to discover which version of EEM is supported on your device. The current latest version of EEM is 4.0.

Writing EEM Policies

As mentioned earlier, EEM offers the ability to monitor events and take informational or corrective action when monitored events occur or a threshold is reached. An EEM policy is an entity that defines an event and the actions to be taken when that event occurs. There are two types of EEM policies: an **applet** or a **script**. An applet is a simple form of policy that is defined in the CLI configuration. A script is a form of policy that is written in Tcl.

The version of EEM is based on the IOS release cycle. It is important to understand the functionality of EEM based on the IOS image so that you fully understand the features and functionality of the event detectors and what is supported.

EEM Applet

Using an EEM applet is a concise method for defining event screening criteria and the actions to be taken when a particular event occurs. In applet configuration mode, three types of configuration statements are supported. The **event** commands are used to specify the event criteria that trigger the applet to run, the **action** commands are used to specify an action to perform when the EEM applet is triggered, and the **set** command is used to set the value of an EEM applet variable.

Only one **event** configuration command is allowed in an applet configuration. When applet configuration mode is exited and the **no event** command is present, a warning is displayed, stating that no event is associated with the applet. If **no event** is specified, the applet is not considered registered. When no action is associated with the applet, events are still triggered, but no actions are performed. Multiple action configuration commands are allowed in an applet configuration. Use the **show event manager policy registered** command to display a list of registered applets.

Before modifying an EEM applet, you need to be aware that the existing applet is not replaced until you exit applet configuration mode. While you are in applet configuration mode modifying the applet, the existing applet may be executing. It is safe to modify the applet without unregistering it. When you exit applet configuration mode, the old applet is unregistered, and the new version is registered.

The **action** configuration commands are uniquely identified using the **label** argument, which can be any string value. Actions are sorted in ascending alphanumeric key sequence, using the **label** argument as the sort key, and they are run using this sequence.

The EEM schedules and runs policies on the basis of an event specification that is contained within a policy. When applet configuration mode is exited, EEM examines the **event** and **action** commands that are entered and registers the applet to be run when a specified event occurs.

EEM Script

You define scripts by using an ASCII editor and then copying the script to the networking devices to register them with EEM. EEM supports Tcl scripts.

EEM allows you to write and implement your own policies using Tcl. Writing an EEM policy involves several actions:

- Selecting the event for which the policy is run.
- Defining the event detector options associated with logging and responding to the event.
- Choosing the actions to be followed when the event occurs.

Cisco provides enhancements to Tcl in the form of keyword extensions that facilitate the development of EEM policies. The main categories of keywords identify the detected event, the subsequent action, utility information, counter values, and system information.

Writing an EEM Policy Using the Cisco IOS CLI

To register an applet with EEM, you can first define environment variables that the applet will use. EEM environment variables for EEM policies are defined using the EEM **event manager environment** configuration command. By convention, every Cisco EEM environment variable begins with an underscore (_). You can display the EEM environment variables set on a system by using the **show event manager environment** privileged EXEC command. For example, you can create EEM policies that can send emails when an event occurs. The variables could include _email_server, _email_to, _email_from, and _email_cc. These variables could then be used in the applet to generate email notifications when specific events occur.

Once the (optional) variables are defined, you use the **event manager applet** *applet-name* global configuration command to enter EEM configuration mode. You then use the **event** and **action** commands to build the applet. Only one **event** command is allowed in an EEM applet, but multiple **action** commands are permitted.

Some of the EEM events that are available are CLI events, config events, interface events, routing events, syslog events, SNMP events, and track events. EEM actions can be numerous, and it can be helpful to add labels so you can sort them.

Example 3-4 shows the creation of an applet that matches a syslog pattern using a regular expression. When the interface GigabitEthernet2/0 changes state to down, an event trigger is generated. The event in this case is to generate a syslog message and email the network administrator that there is an issue with the interface. Notice that environment variables are defined for use in the applet.

Example 3-4 EEM Applet Example

```
Router(config)# event manager environment _email_to engineering@example.com
Router(config)# event manager environment _email_server 192.168.1.10
Router(config)# event manager environment _email_from router@example.com
Router(config)# event manager applet MyApplet
Router(config-applet)# event syslog pattern "Interface GigabitEthernet2/0,
  changed
 state to down"
Router(config-applet)# action 1.0 syslog priority critical msg "Interface Gi2/0
  is down - notifying network team"
Router(config-applet)# action 2.0 mail server _email_server to _email_to from
_email_from subject "Interface Down" body "Router R1 interface Gi2/0 is down"
```

Using EEM and Tcl Scripts

Tool Command Language (Tcl), invented in the late 1980s, is a dynamic programming or scripting language, an interpreter, and a C library. Tcl helps users control other applications or utilities by using basic flow control.

Tcl is built into Cisco routers and switches to allow engineers to interact directly with a device by using various Tcl scripts. Tcl is a very useful scripting language that provides many ways to streamline different tasks that can help with day-to-day operations and monitoring of a network. The following are some of the tasks that can be automated by using these scripts:

- Verifying IP and IPv6 reachability by using **ping**
- Verifying IP and IPv6 reachability by using **traceroute**
- Checking interface statistics
- Retrieving SNMP information by accessing MIBs
- Sending email messages containing CLI output from Tcl scripts

Most often, basic Tcl scripts are entered line by line in the Tcl shell. However, for more advanced scripting methods, you can load a script into the flash memory of the device you are working on and execute the script from there.

You can also verify that an image you are using has the Tcl interpreter by just entering the command **tclsh** at the router prompt in privileged mode:

```
Router# tclsh
Router(tcl)#
```

As you can see here, the command prompt changes to show that you are in Tcl mode. Commands entered at the (tcl) prompt will be handled by the Tcl interpreter if it understands them. If a command is not understood, it is passed along to the standard IOS command process.

Example 3-5 shows a simple Tcl script that pings three devices. That script is saved with the name **ping_script.tcl** and is copied to the router's flash memory by using TFTP. An EEM applet called **myping** is then defined that calls the Tcl script. Notice that when the applet is executed, the router pings the three addresses defined in the Tcl script.

Example 3-5 Tcl and EEM Example

```
foreach address {
+>(tcl)# 172.16.10.1
+>(tcl)# 172.16.10.2
+>(tcl)# 172.16.10.3
+>(tcl)# } { ping $address
+>(tcl)#
}
```

```
event manager applet myping
 event none
 action 1.0 cli command "enable"
 action 1.1 cli command "tclsh flash:/ping_script.tcl"
Router# event manager run myping
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.10.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/6 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.10.2 timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/3/5 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.10.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/6 ms
```

Study Resources

For today's exam topics, refer to the following resources for more study.

Resource	**Module, Chapter, or Links**
CCNP and CCIE Enterprise Core ENCOR 350-401 Official Cert Guide	29
CCNP and CCIE Enterprise Core & CCNP Advanced Routing Portable Command Guide	11
Embedded Event Manager Configuration Guide, Cisco IOS Release 15M&T	https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/eem/configuration/15-mt/eem-15-mt-book.html

Review Lab 1

Objective

Today we review some of the configuration and troubleshooting concepts covered in the ENCOR 350-401 exam topics.

Topology

Figure 2-1 shows the network diagram used for this review lab.

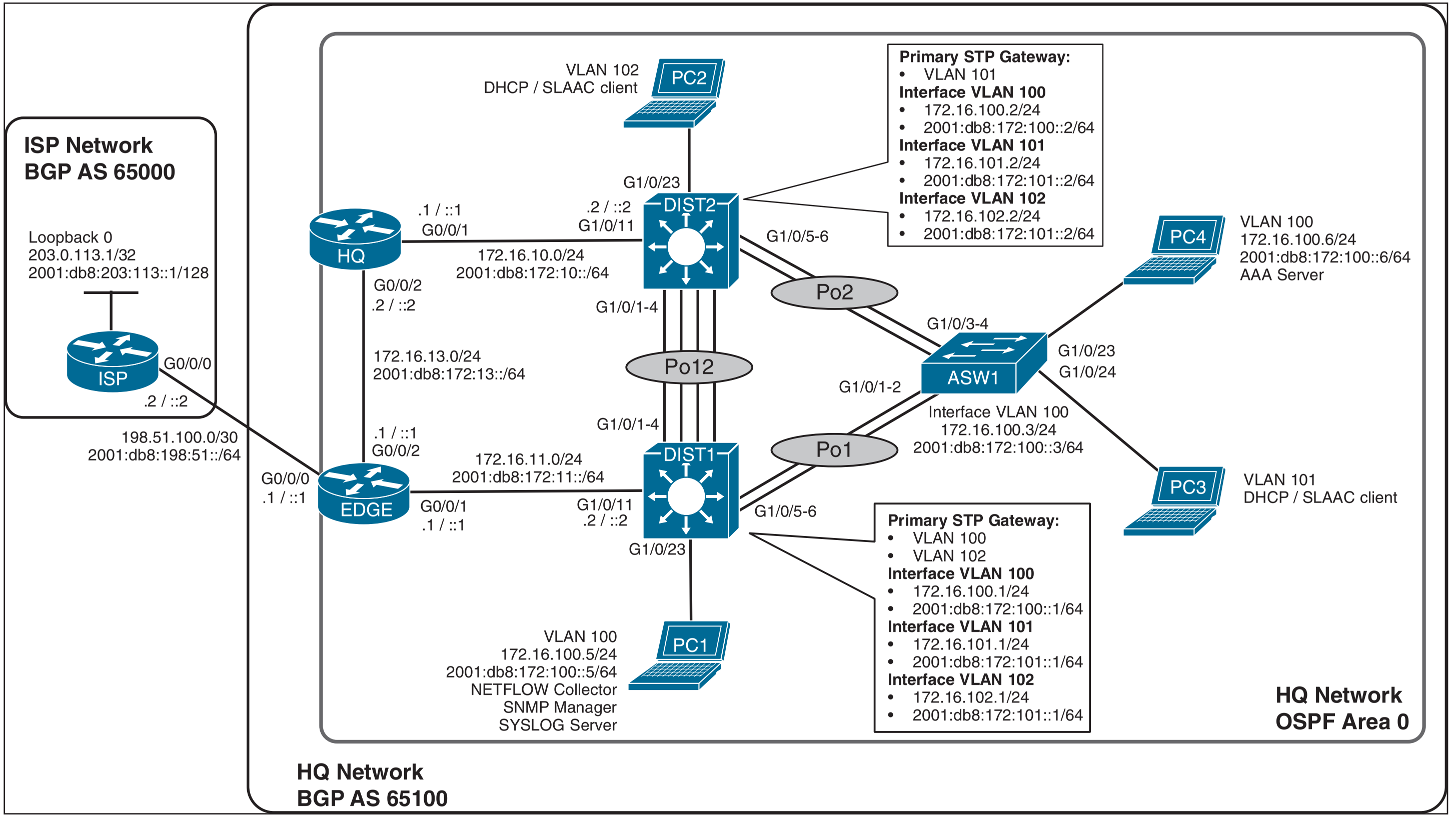
In this lab, you are responsible for completing the configuration of the network so that:

- There is full end-to-end reachability. All PCs should be able to ping Loopback 0 on the ISP router.
- Hosts have reliable default gateway support in the event of a failure at the distribution layer.
- Management protocols such as NetFlow, SNMP, syslog, NTP, and AAA are operational within the HQ Network part of the topology.

Be careful to verify that your configurations meet the provided specifications and that the devices perform as required. Initial configuration scripts and answer scripts are available on the Cisco Press website. You need to register your book to access this bonus content. For information on how to register your book, see the Introduction.

For this review lab, you need three routers (IOS, IOSv, or IOS XE), three switches (Layer 2/Layer 3), and four Windows or Linux PCs.

Figure 2-1 Review Lab 1 Topology Diagram


ISP Network
BGP AS 65000
Loopback 0
203.0.113.1/32
2001:db8:203:113::1/128
ISP
G0/0/0
.2 / ::2
198.51.100.0/30
2001:db8:198:51::/64
G0/0/0
.1 / ::1
HQ Network
BGP AS 65100
HQ Network
OSPF Area 0
HQ
.1 / ::1
G0/0/1
G0/0/2
.2 / ::2
172.16.10.0/24
2001:db8:172:10::/64
172.16.13.0/24
2001:db8:172:13::/64
.1 / ::1
G0/0/2
EDGE
G0/0/1
.1 / ::1
172.16.11.0/24
2001:db8:172:11::/64
VLAN 102
DHCP / SLAAC client
PC2
G1/0/23
.2 / ::2
G1/0/11
DIST2
G1/0/1-4
Po12
G1/0/1-4
DIST1
G1/0/11
.2 / ::2
G1/0/23
PC1
VLAN 100
172.16.100.5/24
2001:db8:172:100::5/64
NETFLOW Collector
SNMP Manager
SYSLOG Server
G1/0/5-6
Po2
G1/0/3-4
ASW1
G1/0/1-2
Po1
G1/0/5-6
Interface VLAN 100
172.16.100.3/24
2001:db8:172:100::3/64
G1/0/23
G1/0/24
PC4
VLAN 100
172.16.100.6/24
2001:db8:172:100::6/64
AAA Server
PC3
VLAN 101
DHCP / SLAAC client
Primary STP Gateway:
• VLAN 101
Interface VLAN 100
• 172.16.100.2/24
• 2001:db8:172:100::2/64
Interface VLAN 101
• 172.16.101.2/24
• 2001:db8:172:101::2/64
Interface VLAN 102
• 172.16.102.2/24
• 2001:db8:172:101::2/64
Primary STP Gateway:
• VLAN 100
• VLAN 102
Interface VLAN 100
• 172.16.100.1/24
• 2001:db8:172:100::1/64
Interface VLAN 101
• 172.16.101.1/24
• 2001:db8:172:101::1/64
Interface VLAN 102
• 172.16.102.1/24
• 2001:db8:172:101::1/64

Addressing Table

Table 2-1 provides the IPv4 and IPv6 addresses for all devices in this review lab.

Table 2-1 Review Lab 1 Addressing Table

Device	Interface	IPv4 Address	IPv6 Address	IPv6 Link-Local Address
EDGE	G0/0/0	198.51.100.1/30	2001:db8:198:51::1/64	fe80::1:1
	G0/0/1	172.16.11.1/24	2001:db8:172:11::1/64	fe80::1:2
	G0/0/2	172.16.13.1/24	2001:db8:172:13::1/64	fe80::1:3
ISP	G0/0/0	198.51.100.2/30	2001:db8:200::2/64	fe80::2:1
	Loopback0	203.0.113.1/32	2001:db8:203:113::1/128	fe80::2:3
HQ	G0/0/1	172.16.10.1/24	2001:db8:172:10::1/64	fe80::3:2
	G0/0/2	172.16.13.2/24	2001:db8:172:13::2/64	fe80::3:3
DIST1	G1/0/11	172.16.11.2/24	2001:db8:172:11::2/64	fe80::d1:1
	VLAN 100	172.16.100.1/24	2001:db8:172:100::1/64	fe80::d1:2
	VLAN 101	172.16.101.1/24	2001:db8:172:101::1/64	fe80::d1:3
	VLAN 102	172.16.101.2/24	2001:db8:172:102::1/64	fe80::d1:4
DIST2	G1/0/11	172.16.10.2/24	2001:db8:172:10::2/64	fe80::d2:1
	VLAN 100	172.16.100.2/24	2001:db8:172:100::2/64	fe80::d2:2
	VLAN 101	172.16.101.2/24	2001:db8:172:101::2/64	fe80::d2:3
	VLAN 102	172.16.102.2/24	2001:db8:172:102::2/64	fe80::d2:4
ASW1	VLAN 100	172.16.100.3/24	2001:db8:172:100::3/64	fe80::a1:1
PC1	NIC	172.16.100.5/24	2001:db8:172:100::5/64	EUI-64
PC2	NIC	DHCP	SLAAC	EUI-64
PC3	NIC	DHCP	SLAAC	EUI-64
PC4	NIC	172.16.100.6/24	2001:db8:172:100::6/64	EUI-64

Tasks

The tasks for this review lab are broken up into six parts:

- **Part 1:** Build the Network and Configure Basic Device Settings and Interface Addressing
- **Part 2:** Configure the Layer 2 Network and Host Support
- **Part 3:** Configure Routing Protocols
- **Part 4:** Configure First-Hop Redundancy and IP SLA Functionality
- **Part 5:** Configure Secure Access
- **Part 6:** Configure Network Management Features

Part 1: Build the Network and Configure Basic Device Settings and Interface Addressing

In Part 1, you set up the network topology and configure basic settings and interface addressing. Follow these steps:

Step 1. Build the network, as shown in the topology in Figure 2-1.

Attach the devices as shown in the topology diagram and connect them as necessary.

Step 2. Configure the basic settings for each device:

- **a.** Console into each device, enter global configuration mode, and apply the basic settings. The startup configurations for each device are provided on the Cisco Press website. After you register your book on the Cisco Press website, you will have access to the text files with the initial configuration for each network device in the topology. You can copy and paste the configuration to each device.
- **b.** Save the running configuration to startup-config on all devices.
- **c.** Configure PC 1 and PC 4 host addressing as shown in the addressing table (refer to Table 2-1). Assign the default gateway address 172.16.100.254, which will be the HSRPv2 virtual IP address used in Part 4.

Part 2: Configure the Layer 2 Network and Host Support

In this part of the review lab, you complete the Layer 2 network configuration and set up basic host support. At the end of this part, all the switches should be able to communicate. PC2 and PC3 should receive addressing from DHCP and SLAAC.

Complete the configuration tasks listed in Table 2-2.

Table 2-2 Part 2 Configuration Tasks

Task #	Task	Specification
2.1	On all switches, configure IEEE 802.1Q trunk interfaces on interconnecting switch links.	Enable 802.1Q trunk links between: ■ DIST1 and DIST2 ■ DIST1 and ASW1 ■ DIST2 and ASW1
2.2	On all switches, change the native VLAN on trunk links.	Use VLAN 999 as the native VLAN.
2.3	On all switches, enable the Rapid Spanning Tree Protocol.	Use Rapid Spanning Tree.
2.4	On DIST1 and DIST2, configure the appropriate RSTP root bridges, based on the information in the topology diagram. DIST1 and DIST2 must provide backup in case of root bridge failure.	Configure DIST1 and DIST2 as root for the appropriate VLANs with mutually supporting priorities in case of switch failure.

Task #	Task	Specification
2.5	On all switches, create LACP EtherChannels, as shown in the topology diagram.	Use the following channel numbers: ■ DIST1 to DIST2: Port channel 12 ■ DIST1 to ASW1: Port channel 1 ■ DIST2 to ASW1: Port channel 2
2.6	On all switches, configure host access ports connecting to PC1, PC2, PC3, and PC4.	Configure access ports with appropriate VLAN settings, as shown in the topology diagram. Host ports should transition immediately to forwarding state.
2.7	Verify IPv4 DHCP services.	PC2 and PC3 are DHCP clients and should be receiving valid IPv4 addresses.
2.8	Verify local LAN connectivity.	PC1 should be able to successfully ping: ■ DIST1: 172.16.100.1 ■ DIST2: 172.16.100.2 ■ PC4: 172.16.100.6 PC2 should be able to successfully ping: ■ DIST1: 172.16.102.1 ■ DIST2: 172.16.102.2 PC3 should be able to successfully ping: ■ DIST1: 172.16.101.1 ■ DIST2: 172.16.101.2 PC4 should be able to successfully ping: ■ D1: 172.16.100.1 ■ D2: 172.16.100.2PC1: 172.16.100.5

Part 3: Configure Routing Protocols

In this part, you configure IPv4 and IPv6 routing protocols. At the end of this part, the network should be fully converged. IPv4 and IPv6 pings to the ISP router's Loopback 0 interface from DIST1 and DIST2 should be successful.

NOTE: Pings from the hosts will not be successful because their default gateways are pointing to the HSRP address, which will be enabled in Part 4.

Complete the configuration tasks listed in Table 2-3.

Table 2-3 Part 3 Configuration Tasks

Task #	Task	Specification
3.1	On the HQ Network (that is, EDGE, HQ, DIST1, and DIST2), configure single-area OSPFv2 in Area 0.	Use OSPF Process ID 4 and assign the following router IDs: ■ EDGE: 0.0.4.1 ■ HQ: 0.0.4.2 ■ DIST1: 0.0.4.3 ■ DIST2: 0.0.4.4 On EDGE, HQ, DIST1, and DIST2, advertise all directly connected networks/VLANs in Area 0. ■ On EDGE, do not advertise the EDGE–ISP network. ■ On EDGE, propagate a default route. Note that the default route will be provided by BGP. Disable OSPFv2 advertisements on: ■ DIST1: All interfaces except G1/0/11 ■ DIST2: All interfaces except G1/0/11
3.2	On the HQ Network (that is, EDGE, HQ, DIST1, and DIST2), configure address family single-area OSPFv3 in Area 0.	Use OSPF Process ID 6 and assign the following router IDs: ■ EDGE: 0.0.6.1 ■ HQ: 0.0.6.2 ■ DIST1: 0.0.6.3 ■ DIST2: 0.0.6.4 On EDGE, HQ, DIST1, and DIST2, advertise all directly connected networks/VLANs in Area 0. ■ On EDGE, do not advertise the EDGE–ISP network. ■ On EDGE, propagate a default route. Note that the default route will be provided by BGP. Disable OSPFv3 advertisements on: ■ DIST1: All interfaces except G1/0/11 ■ DIST2: All interfaces except G1/0/11

Task #	Task	Specification
3.3	On ISP in the ISP Network, configure MP-BGP.	Configure two default static routes via the Loopback 0 interface: ■ An IPv4 default static route ■ An IPv6 default static route Configure ISP in BGP ASN 65000 and use the router ID 2.2.2.2. Configure and enable an IPv4 and IPv6 neighbor relationship with EDGE in ASN 65100. In the IPv4 address family, advertise: ■ The Loopback 0 IPv4 network (/32) ■ The default route (0.0.0.0/0) In the IPv6 address family, advertise: ■ The Loopback 0 IPv6 network (/128) ■ The default route (::/0)
3.4	On EDGE in the HQ Network, configure MP-BGP.	Configure two static summary routes to interface Null 0: ■ A summary IPv4 route for 172.16.0.0/16 ■ A summary IPv6 route for 2001:db8:172::/48 Configure EDGE in BGP ASN 65100 and use the router ID 1.1.1.1. Configure an IPv4 and IPv6 neighbor relationship with ISP in ASN 65000. In IPv4 address family: ■ Disable the IPv6 neighbor relationship. ■ Enable the IPv4 neighbor relationship. ■ Advertise the 172.16.0.0/16 network. In IPv6 address family: ■ Disable the IPv4 neighbor relationship. ■ Enable the IPv6 neighbor relationship. ■ Advertise the 2001:db8:172::/48 network.

Part 4: Configure First-Hop Redundancy and IP SLA Functionality

In this part, you configure HSRP Version 2 to provide first-hop redundancy for hosts in HQ Network and also define multiple IPv4 and IPv6 IP SLAs.

Complete the configuration tasks listed in Table 2-4.

Table 2-4 Part 4 Configuration Tasks

Task #	Task	Specification
4.1	On DIST1, create IP SLAs that test the reachability of EDGE interface G0/0/1.	Create two IP SLAs: ■ Use SLA number 4 for IPv4. ■ Use SLA number 6 for IPv6. The IP SLAs should test the availability of the DIST G0/0/1 interface every 5 seconds. Schedule the SLA for immediate implementation, with no end time. Create an IP SLA object for IP SLA 4 and one for IP SLA 6. ■ Use track number 4 for IP SLA 4. ■ Use track number 6 for IP SLA 6. The tracked objects should notify DIST1 if the IP SLA state changes from down to up after 10 seconds or from up to down after 15 seconds.
4.2	On DIST2, create IP SLAs that test the reachability of HQ interface G0/0/1.	Create two IP SLAs: ■ Use SLA number 4 for IPv4. ■ Use SLA number 6 for IPv6. The IP SLAs should test the availability of the HQ G0/0/1 interface every 5 seconds. Schedule the SLA for immediate implementation, with no end time. Create an IP SLA object for IP SLA 4 and one for IP SLA 6: ■ Use track number 4 for IP SLA 4. ■ Use track number 6 for IP SLA 6. The tracked objects should notify DIST2 if the IP SLA state changes from down to up after 10 seconds or from up to down after 15 seconds.

Task #	Task	Specification
4.3	On DIST1, configure HSRPv2.	DIST1 is the primary router for VLANs 100 and 102; therefore, their priority will be increased to 150. Configure HSRP Version 2. Configure IPv4 HSRP group 104 for VLAN 100: ■ Assign the virtual IP address 172.16.100.254. ■ Set the group priority to 150. ■ Enable preemption. ■ Track object 4 and decrement by 60. Configure IPv4 HSRP group 114 for VLAN 101: ■ Assign the virtual IP address 172.16.101.254. ■ Enable preemption. ■ Track object 4 to decrement by 60. Configure IPv4 HSRP group 124 for VLAN 102: ■ Assign the virtual IP address 172.16.102.254. ■ Set the group priority to 150. ■ Enable preemption. ■ Track object 4 to decrement by 60. Configure IPv6 HSRP group 106 for VLAN 100: ■ Assign the virtual IP address by using **ipv6 autoconfig**. ■ Set the group priority to 150. ■ Enable preemption. ■ Track object 6 and decrement by 60. Configure IPv6 HSRP group 116 for VLAN 101: ■ Assign the virtual IP address by using **ipv6 autoconfig**. ■ Enable preemption. ■ Track object 6 and decrement by 60. Configure IPv6 HSRP group 126 for VLAN 102: ■ Assign the virtual IP address by using **ipv6 autoconfig**. ■ Set the group priority to 150. ■ Enable preemption. ■ Track object 6 and decrement by 60.

Task #	Task	Specification
	On DIST2, configure HSRPv2.	DIST2 is the primary router for VLAN 101; therefore, the priority will be increased to 150. Configure HSRP Version 2. Configure IPv4 HSRP group 104 for VLAN 100: ■ Assign the virtual IP address 172.16.100.254. ■ Enable preemption. ■ Track object 4 and decrement by 60. Configure IPv4 HSRP group 114 for VLAN 101: ■ Assign the virtual IP address 172.16.101.254. ■ Set the group priority to 150. ■ Enable preemption. ■ Track object 4 to decrement by 60. Configure IPv4 HSRP group 124 for VLAN 102: ■ Assign the virtual IP address 172.16.102.254. ■ Enable preemption. ■ Track object 4 to decrement by 60. Configure IPv6 HSRP group 106 for VLAN 100: ■ Assign the virtual IP address by using **ipv6 autoconfig**. ■ Enable preemption. ■ Track object 6 and decrement by 60. Configure IPv6 HSRP group 116 for VLAN 101: ■ Assign the virtual IP address by using **ipv6 autoconfig**. ■ Set the group priority to 150. ■ Enable preemption. ■ Track object 6 and decrement by 60. Configure IPv6 HSRP group 126 for VLAN 102: ■ Assign the virtual IP address by using **ipv6 autoconfig**. ■ Enable preemption. ■ Track object 6 and decrement by 60.

Part 5: Configure Secure Access

In this part you configure various security mechanisms on the devices in the topology.

You need to set up a AAA RADIUS server on PC4 for this part of the lab. The shared secret key should be configured with 31daysRADIUS, and the server should use UDP ports 1812 and 1813. Create a user called 31days with the password 31dayspass.

Complete the configuration tasks listed in Table 2-5.

Table 2-5 Part 5 Configuration Tasks

Task #	Task	Specification
5.1	On every Cisco network device, secure privileged EXEC using the SCRYPT encryption algorithm.	Password: 31daysENCOR
5.2	On every Cisco network device, create a local user and secure it using the SCRYPT encryption algorithm.	SCRYPT encrypted account specifics: ■ Local username: admin ■ Privilege level: 15 ■ Password: 31daysENCOR
5.3	On every Cisco network device (except ISP), enable AAA.	Enable AAA.
5.4	On every Cisco network device (except ISP), configure the RADIUS server specifics.	RADIUS server specifics: ■ RADIUS server IP address: 172.16.100.6 ■ RADIUS server UDP ports: 1812 and 1813 ■ Password: 31days RADIUS
5.5	On every Cisco network device (except ISP), configure the AAA authentication method list.	AAA authentication specifics: ■ Use the default method list. ■ Validate against the RADIUS server group. ■ Otherwise, use the local database.
5.6	Verify the AAA service on every Cisco network device (except ISP).	Log out and log in to all devices (except ISP), using the username 31days and the password 31dayspass. You should be successful.

Part 6: Configure Network Management Features

In this part, you configure various network management features.

You need to set up the following management services on PC1 for this part of the lab:

- PC1 should have a syslog server running to receive messages from the HQ network devices.
- PC1 should be configured to run an SNMP trap receiver. Use the community string 31DAYS.
- PC1 should have a NetFlow Version 9 collector running on port 9999 to received data exported from the EDGE router.

Complete the configuration tasks listed in Table 2-6.

Table 2-6 Part 6 Configuration Tasks

Task #	Task	Specification
6.1	On every Cisco network device, set the local clock to the current UTC time.	Set the local clock to the current UTC time.
6.2	Configure ISP as an NTP master.	Configure ISP as an NTP master at stratum level 3.
6.3	Configure NTP on EDGE, HQ, DIST1, DIST2, and ASW1.	Configure NTP as follows: ■ EDGE must synchronize with ISP. ■ HQ DIST1 and ASW1 should synchronize time with EDGE. ■ DIST2 should synchronize time with HQ.
6.4	Configure syslog on every Cisco network device except ISP.	Syslog messages should be sent to PC1 at 172.16.100.5 at the WARNING level.
6.5	Configure SNMPv2c on every Cisco network device except ISP.	SNMPv2 specifics: ■ Only read-only SNMP will be used. ■ Limit SNMP access to PC1's IP address. ■ Configure the SNMP contact value to your name. ■ Set the community string to 31DAYS. ■ On HQ, DIST1, and DIST2, enable traps config and ospf to be sent. ■ On EDGE, enable traps bgp, config, and ospf to be sent. ■ On ASW1, enable traps config to be sent.
6.6	Optional (IOS routers only): Configure NetFlow on the EDGE router. Note that IOS XE devices only support Flexible NetFlow.	Enable NetFlow Version 9 on the EDGE router. Track flows entering and exiting interface G0/0/0 on that router and send NetFlow information to PC1 using port 9999.

Day 1

Review Lab 2

Objective

Today we continue to review configuration and troubleshooting concepts covered in the ENCOR 350-401 exam topics.

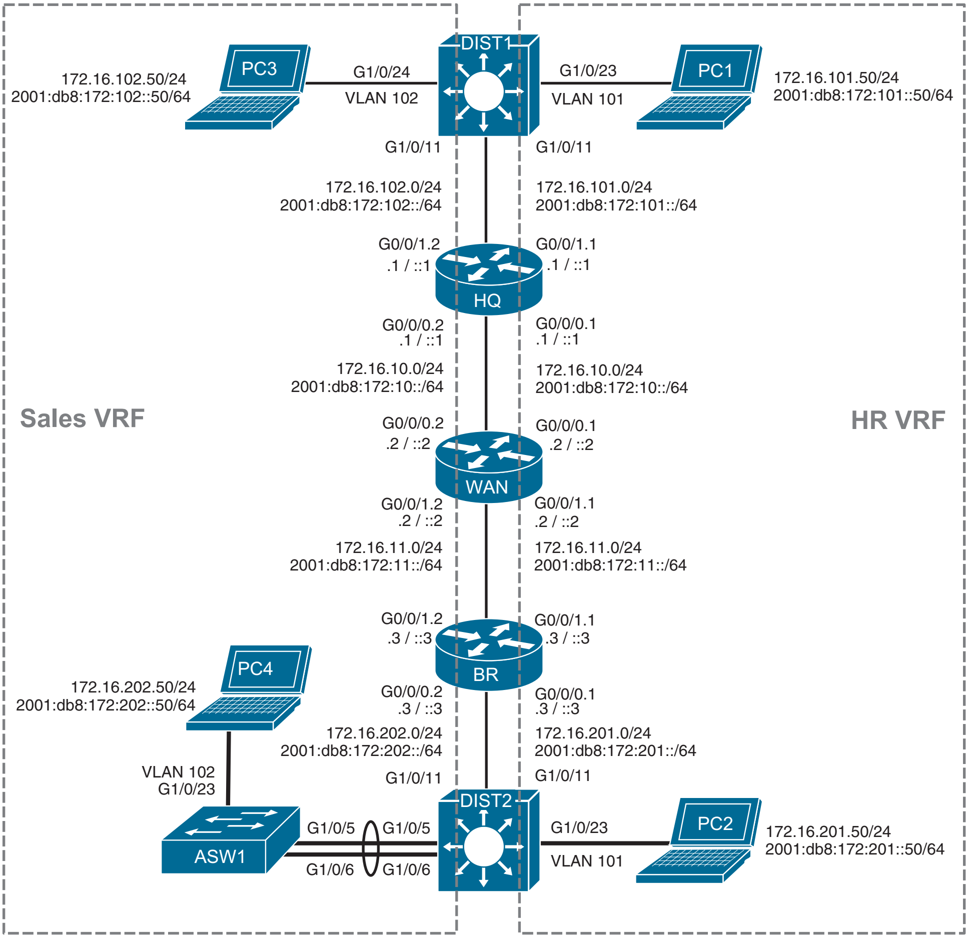
Topology

Figure 1-1 shows the network diagram used for this review lab.

In this lab, you are responsible for completing the multi-VRF configuration of the network supporting sales users and HR users. Upon completion, there should be full end-to-end reachability, and the two groups should not be able to communicate with each other. Be careful to verify that your configurations meet the provided specifications and that the devices perform as required. Initial configuration scripts and answer scripts are available on the Cisco Press website. You need to register your book to access this bonus content. For information on how to register your book, see the Introduction.

For this review lab, you need three routers (IOS, IOSv, or IOS XE), three switches (Layer 2/ Layer 3), and four Windows or Linux PCs.

Figure 1-1 Review Lab 2 Topology Diagram


Sales VRF
HR VRF
DIST1
PC3
172.16.102.50/24
2001:db8:172:102::50/64
G1/0/24
VLAN 102
PC1
172.16.101.50/24
2001:db8:172:101::50/64
G1/0/23
VLAN 101
G1/0/11
G1/0/11
172.16.102.0/24
2001:db8:172:102::/64
172.16.101.0/24
2001:db8:172:101::/64
G0/0/1.2
.1 / ::1
G0/0/1.1
.1 / ::1
HQ
G0/0/0.2
.1 / ::1
G0/0/0.1
.1 / ::1
172.16.10.0/24
2001:db8:172:10::/64
172.16.10.0/24
2001:db8:172:10::/64
G0/0/0.2
.2 / ::2
G0/0/0.1
.2 / ::2
WAN
G0/0/1.2
.2 / ::2
G0/0/1.1
.2 / ::2
172.16.11.0/24
2001:db8:172:11::/64
172.16.11.0/24
2001:db8:172:11::/64
G0/0/1.2
.3 / ::3
G0/0/1.1
.3 / ::3
BR
G0/0/0.2
.3 / ::3
G0/0/0.1
.3 / ::3
172.16.202.0/24
2001:db8:172:202::/64
172.16.201.0/24
2001:db8:172:201::/64
G1/0/11
G1/0/11
PC4
172.16.202.50/24
2001:db8:172:202::50/64
VLAN 102
G1/0/23
ASW1
G1/0/5
G1/0/5
G1/0/6
G1/0/6
DIST2
G1/0/23
VLAN 101
PC2
172.16.201.50/24
2001:db8:172:201::50/64

Addressing Table

Table 1-1 provides the IPv4 and IPv6 addresses for all devices in this review lab.

Table 1-1 Review Lab 2 Addressing Table

Device	Interface	IPv4 Address	IPv6 Address	IPv6 Link-Local Address
HQ	G0/0/0.1	172.16.10.1/24	2001:db8:172:10::1/64	fe80::1:1
	G0/0/0.2	172.16.10.1/24	2001:db8:172:10::1/64	fe80::1:2
	G0/0/1.1	172.16.101.1/24	2001:db8:172:101::1/64	fe80::1:3
	G0/0/1.2	172.16.102.1./24	2001:db8:172:102::1/64	Fe80::1:4

Device	Interface	IPv4 Address	IPv6 Address	IPv6 Link-Local Address
WAN	G0/0/0.1	172.16.10.2/24	2001:db8:172:10::2/64	fe80::2:1
	G0/0/0.2	172.16.10.2./24	2001:db8:172:10::2/64	fe80::2:2
	G0/0/1.1	172.16.11.2/24	2001:db8:172:11::2/64	fe80::2:3
	G0/0/1.2	172.16.11.2/24	2001:db8:172:11::2/64	fe80::2:4
BR	G0/0/0.1	172.16.201.3/24	2001:db8:172:201::3/64	fe80::3:1
	G0/0/0.2	172.16.202.3/24	2001:db8:172:202::3/64	fe80::3:2
	G0/0/1.1	172.16.11.3/24	2001:db8:172:11::3/64	fe80::3:3
	G0/0/1.2	172.16.11.3/24	2001:db8:172:11::3/64	fe80::3:4
PC1	NIC	172.16.101.50/24 DG: 172.16.101.1/24	2001:db8:172:101::50/64 DG: 2001:db8:172:101::1/64	EUI-64
PC2	NIC	172.16.201.50/24 DG: 172.16.201.3/24	2001:db8:172:201::50/64 DG: 2001:db8:172:201::3/64	EUI-64
PC3	NIC	172.16.102.50/24 DG: 172.16.102.1/24	2001:db8:172:102::50/64 DG: 2001:db8:172:102::1/64	EUI-64
PC4	NIC	172.16.202.50/24 172.16.202.3/24	2001:db8:172:202::50/64 DG: 2001:db8:172:202::3/64	EUI-64

Tasks

The tasks for this review lab are broken up into four parts:

- **Part 1:** Build the Network and Configure Basic Device Settings
- **Part 2:** Configure VRF and Static Routing
- **Part 3:** Configure Layer 2 Network
- **Part 4:** Configure Secure Access

Part 1: Build the Network and Configure Basic Device Settings

In Part 1, you set up the network topology and configure basic settings. Follow these steps:

Step 1. Build the network as shown in the topology in Figure 1-1.

Attach the devices as shown in the topology diagram and connect them as necessary.

Step 2. Configure the basic settings for each device:

a. Console into each device, enter global configuration mode, and apply the basic settings. The startup configurations for each device are provided on the Cisco Press website. After you register your book on the Cisco Press website, you will have access to the text files with the initial configuration for each network device in the topology. You can copy and paste the configuration to each device.

b. Save the running configuration to startup-config on all devices.

c. Configure PC 1 and PC 4 host addressing as shown in the addressing table (refer to Table 1-1).

Part 2: Configure VRF and Static Routing

In this part of the review lab, you will configure VRF-Lite on all three routers and also configure the appropriate static routes to support end-to-end reachability. At the end of this part, R1 should be able to ping R3 in each VRF instance.

Complete the configuration tasks listed in Table 1-2.

Table 1-2 Part 2 Configuration Tasks

Task #	Task	Specification
2.1	On HQ, WAN, and BR, configure VRF-Lite VRF instances, as shown in the topology diagram (refer to Figure 1-1).	Configure two VRF instances: ■ HR-Users ■ Sales-Users The VRF instances must support IPv4 and IPv6.
2.2	On HQ, WAN, and BR, configure IPv4 and IPv6 interfaces on each VRF instance as detailed in the addressing table (refer to Table 1-1).	All routers will use router-on-a-stick on their G0/0/x interfaces to support separation of the VRFs. Subinterface 1, the HR Users VRF instance: ■ Use dot1q encapsulation 101 ■ IPv4 and IPv6 GUA and link-local addresses ■ Enable the interfaces Subinterface 2, the Sales Users VRF instance: ■ Use dot1q encapsulation 102 ■ IPv4 and IPv6 GUA and link-local addresses ■ Enable the interfaces
2.3	On HQ, WAN, and BR, configure the appropriate static routes. Hint: You need two IPv4 and two IPv6 static routes on both HQ and BR routers, but you need four IPv4 and four IPv6 static routes on WAN to ensure end-to-end connectivity. Hint: You can use default routes on HQ and BR.	Configure VRF static routes for both IPv4 and IPv6 in both VRF instances.

Task #	Task	Specification
2.4	Verify connectivity in each VRF instance.	From HQ, verify connectivity to BR: ping vrf Sales-Users 172.16.202.3 ping vrf Sales-Users 2001:db8:172:202::3 ping vrf HR-Users 172.16.201.3 ping vrf HR-Users 2001:dbb:172:201::3

Part 3: Configure Layer 2 Network

In this part, you configure the switches to support host connectivity.

Complete the configuration tasks listed in Table 1-3.

Table 1-3 Part 3 Configuration Tasks

Task #	Task	Specification
3.1	On DIST, DIST2, and ASW1, disable all interfaces.	On all switches, shut down G1/0/1 to G1/0/24.
3.2	On DIST1 and DIST2, configure the trunk links to HQ and BR.	Configure and enable the G1/0/11 link as a trunk link.
3.3	On DIST1 and ASW1, configure the EtherChannel.	On DIST1 and ASW1: ■ Configure and enable Interface G1/0/5 and G1/0/6. ■ Configure and enable Port Channel 1 using PAgP. ■ Enable trunking.
3.4	On DIST1, DIST2, and ASW1, configure access ports for PC1, PC2, PC3, and PC4.	Configure and enable the access ports as follows: ■ On DIST1, configure interface G1/0/23 as an access port in VLAN 101 and enable PortFast. ■ On DIST1, configure interface G1/0/24 as an access port in VLAN 102 and enable PortFast. ■ On DIST2, configure interface G1/0/23 as an access port in VLAN 101 and enable PortFast. ■ On ASW1, configure interface G1/0/23 as an access port in VLAN 102 and enable PortFast.
3.5	Verify PC to PC connectivity.	From PC1, verify IPv4 and IPv6 connectivity to PC2. From PC3, verify IPv4 and IPv6 connectivity to PC4.

Part 4: Configure Secure Access

In this part, you configure various security mechanisms on the devices in the topology.

Complete the configuration tasks listed in Table 1-4.

Table 1-4 Part 4 Configuration Tasks

Task #	Task	Specification
4.1	On every Cisco network device, secure privileged EXE mode.	Configure an enable secret as follows: ■ Algorithm type: SCRYPT ■ Password: 31daysencor
4.2	On every Cisco network device, create a local user account.	Configure a local user: ■ Name: admin ■ Privilege level: 15 ■ Algorithm type: SCRYPT ■ Password: 31daysencor
4.3	On every Cisco network device, enable AAA and enable AAA authentication.	Enable AAA authentication using the local database on all lines using the default method list.

Index

Symbols

- **# (pound sign), Ansible playbooks, 605**

Numbers

- **2-way state, OSPF neighbors, 128**
- **3DES algorithm, IPsec, 237**
- **4xx HTTP status code, 586**
- **5xx HTTP status code, 586**
- **200 (OK) HTTP status code, 586**
- **202 (Accepted) HTTP status code, 586**
- **802.11 Wireless QoS: 802.11e, 336–337**
- **802.11a wireless standard, 400**
- **802.11ac wireless standard, 400**
  - MIMO, 401–402
    - *beamforming, 403–404*
    - *maximal-ratio combining, 402–403*
    - *spatial multiplexing, 402–403*
  - MU-MIMO, 405–406
- **802.11ax (Wi-Fi 6) wireless standard, 401**
- **802.11b wireless standard, 399–400**
- **802.11g wireless standard, 400**
- **802.11n MIMO, 400–402**
  - beamforming, 403–404
  - maximal-ratio combining, 402–403
  - spatial multiplexing, 402–403
- **802.1p Class of Service, 334–336**
- **802.1Q trunk ports, 38–39**
  - configuring, 41
  - tagging process, 39–40
  - verifying, 41–43
- **802.1X wireless standard**
  - endpoint authentication, 485–486
- **user authentication, 449–451, 453–457**

A

- **AAA framework, 497**
  - accounting, 495, 503
  - authentication, 495, 496, 498–499
  - authorization, 495, 501–503
  - enabling, 499–500
  - flexibility, 496
  - granular access control, 496
  - local users, configuring, 499–500
  - overview, 495–496
  - RADIUS, 496–497
    - *configuring, 500–501*
    - *packet flow, 497–498*
  - redundancy, 496
  - scalability, 496
  - TACACS+, 497
    - *configuring, 501*
    - *packet flow, 498*
- **ABR (Area Border Routers)**
  - OSPF, 133
    - *route filtering, 152*
    - *route summarization, 148*
  - OSPFv3, 156
- **access**
  - ACL, 504
    - *applying to interfaces, 511–513*
    - *extended ACL, 508, 513*
    - *inbound ACL, 511–512*
    - *named ACL, 508, 510–511*
    - *numbered ACL, 508–510*
    - *outbound ACL, 511*
    - *overview, 504–505*
    - *standard ACL, 507–508, 512*
    - *wildcard masking, 505–507*
  - configuring, AAA framework, granular access control, 496
  - device access security, 492
    - *enable secret password, 492–493*
    - *line passwords, 493–494*
    - *passwords, 494–495*
    - *usernames, 494–495*
  - granular access control, AAA framework, 496
  - guest access with WebAuth, 457–458
    - *auto-anchor mobility (guest tunneling), 458*
    - *centralized web authentication, 459*
    - *configuring WebAuth, 459–463*
    - *local web authentication, 458*
    - *local web portal, external authentication, 459*
  - RBAC, 289
  - SD-Access, 11–12, 269–270
    - *Cisco DNA Center, 270–272*
    - *fabric technology, 269, 272–281*
    - *overview, 272–274*
    - *WLAN deployments, 408*

- secure access
  - *control, 491*
  - *Review Lab 1, 630–631*
  - *Review Lab 2, 637–638*
- SGT, 487

access--distribution interconnection designs, common, 11
- Layer 2
  - *looped design, 10*
  - *loop-free design, 10*
- Layer 3 access layer (router-based) design, 11
- VSS, 11

access layer (hierarchical LAN design), 2–3
- Layer 2 access layer (STP-based) design, 6–7
  - *looped design, 10*
  - *loop-free design, 10*
- Layer 3 access layer (router-based) design, 7–8, 11
- simplified campus design, 9–10
- software-defined access (SD-access) design, 11–12

access lists
- debug command, 359–360
- NTP, 222–223

access ports, VLAN, 35–36

accounting, AAA framework, 495, 503

accuracy, locations, 444–445

acknowledge packets, 109

ACL (Access Control Lists), 504
- 802.1X endpoint authentication, 487
- extended ACL, 508, 513
- inbound ACL, 511–512
- interfaces, applying to, 511–513
- Layer 2 switches, 17
- Layer 3 switches, 18
- named ACL, 508, 510–511
- numbered ACL, 508–510
- outbound ACL, 511
- overview, 504–505
- standard ACL, 507–508, 512
- wildcard masking, 505–507

active mode (LACP), 94

Active neighbor state, BGP, 171

active scanning, wireless roaming, 435

Active state, HSRP, 191

Ad hoc blocks, multicast addressing, 309

addressing
- group addresses, multicasting, 307–309
- IPv4 addressing
  - *multicasting, group addresses, 307–309*
  - *WLC discovery process, 422*
- IPv6 addressing, WLC discovery process, 422
- multicasting
  - *Ad hoc blocks, 309*
  - *Administratively Scoped blocks (239.0.0.0/8), 309*
  - *GLOP blocks (233.0.0.0/8), 309*
  - *group addresses, 307–309*
  - *Internetwork Control blocks (224.0.1/24), 309*
  - *Local Network Control blocks (224.0.0/24), 308*
  - *SDP/SAP blocks (224.2.0.0/16), 309*
  - *Source-Specific Multicast blocks (232.0.0.0/8), 309*
- NAT, 207–208
- private IP addresses, SD-WAN, 297
- public IP addresses, SD-WAN, 297
- RLOC addresses, 252–253
- system IP addresses, SD-WAN, 296
- tables
  - *Review Lab 1, 623*
  - *Review Lab 2, 634–635*

administration, SD-WAN, 591

Administratively Scoped blocks (239.0.0.0/8), 309

advanced malware protection, firewalls, 478

advertisements, VTP, 47

AES, IPsec, 237–238

agent-based configuration, 599

agentless configuration, 599

aggregating ports, EtherChannel, 91
- configuring, 95–96
- LACP, 93–94
  - *configuring, 97–98*
  - *fast rate timer, 103–104*
  - *hot-standby ports, 101–102*
  - *max bundle feature, 102–103*
  - *port channel min-links, 103*
  - *port priority, 103*
  - *system priority, 103*

  load balancing, 96–97
  need for, 91–93
  PAgP, 93–95, 98–99
  verifying, 97, 99–101

aggregating routes, BGP, 175

aggregator attribute, BGP, 176

aggressive mode (UDLD), 82

AH (Authentication Headers), 236

AI-Driven Data, DNA Center Assurance, 550–551

alerts, retrospective alerting (cloud recall), 475

algorithms
  3DES algorithm, 237
  AES, 237–238
  DES algorithm, 237
  RSA, 237
  symmetric encryption algorithms, 237–238

allowed VLAN, 40–41

ALT routers, 255

alternate ports, 64

AMP for Networks, 475

amplitude, RF communications, 391–392

analytics
  CMX analytics tools, 444
  DNA Center, 536
  location analytics, 444
  SD-WAN, 293–294

Ansible
  Ansible Tower, 602
  control station, 602
  inventory files, 602, 605–606
  modules, 602, 604
  operations, 603–604
  playbooks, 602, 604
    components, 604–605
    example, 604–605
    executing, 606
    inventory files, 605–606
    pound sign (#), 605
    terms, 604
  plays, 604
  push model, 603–604
  roles, 604
  system management tools, 603
  tasks, 604
  YAML files, 602

antennas
  beamforming, 403–404
  directional antennas, 430–431
  EIRP, 398–399
  isotropic antennas, 397–399
  omnidirectional antennas, 429–430
  RF communications, 397–399
  types, 428–431

antireplay protection, IPsec, 237

antispyware software, 472

antivirus software, 472

Anycast Layer 3 gateways, 278

AP (Access Points)
  802.11ac MIMO, 401–402
    beamforming, 403–404
    maximal-ratio combining, 402–403
  802.11ac MU-MIMO, 405–406
  802.11n MIMO, 400–402
    beamforming, 403–404
    maximal-ratio combining, 402–403
  autonomous deployments, 408–410
  bridge mode, 427
  CAPWAP, 413
  centrally switched mode, 426
  Cisco Mobility Express
    master AP, 422
    subordinate AP, 422
  connected mode, 426–427
  fabric mode AP, 277, 279–281
  failovers, 423–424
    AP Fallback, 424–425
    prioritizing, 424
  Fallback, 424–425
  FlexConnect mode, 425–427
  free path loss, 392–393
  intra-controller traffic flows, 414–415
  join orders, 423
  local mode, 425
  locally switched mode, 426
  monitor mode, 428
  OEAP mode, 427–428
  rogue detector mode, 428
  SE-Connect mode, 428
  sniffer mode, 428
  split MAC, centralized WLC deployments, 411–413
  standalone mode, 427
  wireless-to-wired AP traffic flows, 409
  wireless-to-wireless AP traffic flows, 410

- WLC discovery process, 422
    - *AP failovers, 423–425*
    - *AP join orders, 423*
    - *AP modes, 425–428*
    - *DHCP, 423*
    - *DNS, 423*
    - *IPv4 addressing, 422*
    - *IPv6 addressing, 422*
    - *subnetwork broadcasts, 422*

API (Application Programming Interfaces), 583
- application plane, 583
- architectures, 583–584
- Command Runner API, 589
- control plane, 583
- data (forwarding) plane, 583
- File API, 589
- integration API, DNA Center, 590
- Intent API, Data Center, 585–586, 588
- Network Discovery API, 589
- northbound API, 584–585
- Path Trace API, 589
- PnP API, 588
- REST API, 584–585
    - *authentication, 589*
    - *communication process, 585*
    - *CRUD operations, 585–586*
    - *DNA Center, 587–590*
    - *HTTP status codes, 586*
    - *integration API, 590*
    - *Know Your Network REST request paths, 588*
    - *multivendor support, 589–590*
    - *operational tools, 589*
    - *response codes, 585–586*
    - *SD-WAN, 590–598*
    - *security, 586–587*
    - *site management API, 588*
    - *vManage, 590–598*
- site management API, 588
- southbound API, 584
- SWIM API, 588
- Task API, 589
- Template Programmer API, 589

applets, EEM policies, 616–619

Application Health, DNA Center Assurance, 552–554

applications
- API, 583
    - *application plane, 583*
    - *architectures, 583–584*
    - *control plane, 583*
    - *data (forwarding) plane, 583*
    - *Intent API, Data Center, 585–586, 588*
    - *northbound API, 584–585*
    - *REST API, 585–598*
    - *southbound API, 584*
- application sets, 347
- DNA Center application policies, 347–348
- multicasting, 306–307
- NBAR, 331–332
- SD-WAN
    - *application performance optimization, 294–295*
    - *application-aware routing, 294*
- visibility and control, firewalls, 478

architectures
- API, 583–584
- campus networks
    - *Clos architectures, 12–13*
    - *Layer 2 access layer (STP-based) design, 6–7*
    - *Layer 2 looped design, 10*
    - *Layer 2 loop-free design, 10*
    - *Layer 3 access layer (router-based) design, 7–8, 11*
    - *simplified campus design, 9–11*
    - *software-defined access (SD-access) design, 11–12*
    - *spine and leaf architectures, 12–13*
    - *StackWise, simplified campus design, 9–10*
    - *three-tier design, 5–6*
    - *two-tier (collapsed core) design, 4–5*
    - *VSS, simplified campus design, 9–11*
- EEM, 615
- enterprise networks, 4
    - *Clos architectures, 12–13*
    - *Layer 2 access layer (STP-based) design, 6–7*
    - *Layer 2 looped design, 10*
    - *Layer 2 loop-free design, 10*
    - *Layer 3 access layer (router-based) design, 7–8, 11*
    - *simplified campus design, 9–11*
    - *software-defined access (SD-access) design, 11–12*
    - *spine and leaf architectures, 12–13*
    - *StackWise, simplified campus design, 9–10*
    - *three-tier design, 5–6*

        two-tier (collapsed core) design, 4–5
        VSS, simplified campus design, 9–11
    Enterprise NFV, 523–524
    IGMP, 311
    IT4IT reference architecture, 590
    Puppet, 607
    SaltStack, 612–614
    split MAC, centralized WLC deployments, 411–412

Area ID, hello packets, 126

area ranges, OSPF route filtering, 153

ARP (Address Resolution Protocol) tables, CEF, 21, 24

arrays, SD-WAN, 591

AS_path attribute, BGP, 175

ASBR (Autonomous System Boundary Routers)
    OSPF, 133, 149
    OSPFv3, 156

assurance, network
    basic steps, 350
    concepts, 349–350
    connectivity testing, 353, 356–357
    debug command, 357
        access lists, 359–360
        conditional debug command, 358–359
    diagnostics
        debug command, 357–360
        ping command, 352–355
        principles, 350–351
        traceroute command, 355–357
    DNA Center Assurance, 541–542, 548–550
        AI-Driven Data, 550–551
        Application Health, 548–550
        Baseline Deviation Reporting, 551
        Client 360, 551–552
    IOS IP SLA, 360–365
    logging services, 373–374
        Flexible NetFlow, 384–387
        NetFlow, 378–387
        SNMP, 376–378
        syslog, 374–376
    ping command, 352
        connectivity testing, 353
        extended ping command, 353–355
        options, 353
    process, 351–352
    SPAN
        ERSPAN, 370–372
        local SPAN, 366–368
        overview, 365–366
        remote SPAN, 368–370
    traceroute command, 355–357
    traffic sniffers, 365

atomic aggregate path attribute, BGP, 175

attacks, network security, 466–468

authentication
    802.1X user authentication, 449–451, 453–457
    AAA framework, 495, 496, 498–499
    AH, 236
    DNA Center, 589
    EAP, 452–457
    ECDSA signatures, 239
    EIGRP, 110
    endpoints, 802.1X, 485–486
    external authentication, local web portal, 459
    hello packets, 126
    HMAC, 238
    HSRP, 195–196
    local web authentication, 458
    MAB, 487–488
    NTP, 222
    origin authentication, IPsec, 237
    PSK, 238
    PSK authentication, 446–449
    RSA encrypted nonces, 239
    RSA signatures, 238
    VRRP, 201
    WebAuth, 489–490
        device registration, 490
        EasyConnect, 490
        NAD with Central WebAuth, 489
        Wired NAD with Local WebAuth, 489
    WebAuth, guest access with, 457–458
        auto-anchor mobility (guest tunneling), 458
        centralized web authentication, 459
        configuring WebAuth, 459–463
        local web authentication, 458
        local web portal, external authentication, 459
    wireless client authentication
        802.1X user authentication, 449–451, 453–457
        EAP, 452–457
        guest access with WebAuth, 457–463

        PSK authentication, 446–449
        WPA, 445–449

authorization
    802.1X endpoint authentication, 487
    AAA framework, 495, 501–503

Auto mode (PAgP), 95

auto-anchor mobility (guest tunneling), 441–442, 458

automation
    configuration management tools (DevOps), 599–601
        Chef, 600–601
        Puppet, 600
        SaltStack, 601
    Day N network automation, 547–548
    DNA Center, 536, 542–548
    SD-WAN, 293–294
    system management
        Ansible, 602–606
        Chef, 610–612
        EEM, 615–620
        Puppet, 607–610
        SaltStack, 612–615

autonomous AP deployments, 408–410

AS (Autonomous Systems), 122–123
    AS numbers, 109, 163
    OSPF (Open Shortest Path First), 133

AutoQoS, 347

auto-RP, 320–321

AUX ports, syslog messages, 374

avoidance, congestion, 344–345

B

backbone (area 0) areas, OSPF, 133

backbone routers, OSPF, 133

background scanning, wireless roaming, 435

backup ports, 64

bandwidth
    converged networks, 324
    EIGRP, reduced bandwidth use, 106
    multicasting, 306
    OSPF bandwidth, adjusting, 143–144
    unicasting, 306

Bandwidth (K1) metric, EIGRP, 110

bare-metal hypervisors, 521

Baseline Deviation Reporting, DNA Center, 551

batched updates, BGP, 165

BDR (Backup Designated Routers)
    hello packets, 126
    OSPF BDR election process, 134–136

beacons (monitor), SaltStack, 613

beamforming, 403–404

being Pythonic (idiomatic Python), 558–559

best effort QoS model, 327, 328

BGP (Border Gateway Protocol), 163
    aggregator attribute, 176
    AS numbers, 163
    batched updates, 165
    community attribute, 176
    configuring, 176–182
    dual multihomed connections, 168
    dual-homed connections, 168
    EBGP, 172–173
        configuring, 176–178
        verifying, 178–182
    EGP, 164
    hop-by-hop path selection, 166–167
    IBGP, 173
    IGP, 164
    interdomain routing, 163–164
    IP routing tables, 169
    keepalive messages, 164
    MED attribute, 176
    messages
        KEEPALIVE messages, 170
        NOTIFICATION messages, 171
        OPEN messages, 169–170
        UPDATE messages, 170
    multihoming, 167–168
    neighbors
        neighbor states, 171
        neighbor tables, 169
        relationships, 172
    operations, 169
    optional attributes
        nontransitive attributes, 176
        transitive attributes, 176
    path attributes, 175
    path selection, 173–174
    path vectors, 165–166
    performance speed, 165
    route aggregation, 175
    route selection, 174
    routing, 166–167
    single-homed connections, 168

- tables, 169
  - *IP routing tables, 169*
  - *neighbor tables, 169*

BID (Bridge ID), 59, 60

BIDIR-PIM mode, 317

blocking state (ports), 63

Boolean data type, Python, 559, 562

border node, fabric devices, 277, 278

BPDU (Bridge Protocol Data Units), 57
- BPDU Guard, 77–79
- Configuration BPDU, 57
- ports and BPDU frames, 65
- TCA BPDU, 57
- TCN BPDU, 57, 58
- topology changes, 58
  - *direct topology changes, 58–59*
  - *indirect topology changes, 59*
  - *insignificant topology changes, 59*

bridge mode, AP, 427

bridges
- ID, verifying RSTP, 69–70
- priorities, 59–60
- root bridges
  - *bridge priorities, 59–60*
  - *elections, 59–60*
  - *Extended System ID, 60*
  - *MAC addresses, 60*
- STP, prioritizing, 71–73

bridging loops, STP, 55–56

broadcast networks, 133, 134

broadcast probes, wireless roaming, 435

broadcast storms, redundant topologies, 56

broadcast/multicast mode, NTP, 221

BSR (Bootstrap Routers), PIM, 320–321

buffer overflow attacks, 467

C

calculating metrics (K values), 110–111

CAM tables
- Layer 2 switches, 15–17
- Layer 3 switches, 18–19

campus networks
- architectures
  - *Clos architectures, 12–13*
  - *Layer 2 access layer (STP-based) design, 6–7*
  - *Layer 2 looped design, 10*
  - *Layer 2 loop-free design, 10*
  - *Layer 3 access layer (router-based) design, 7–8, 11*
  - *simplified campus design, 9–11*
  - *software-defined access (SD-access) design, 11–12*
  - *spine and leaf architectures, 12–13*
  - *StackWise, simplified campus design, 9–10*
  - *three-tier design, 5–6*
  - *two-tier (collapsed core) design, 4–5*
  - *VSS, simplified campus design, 9–11*
- local VLAN, 34–35

candidate RP, 320

CAPWAP (Control and Provisioning of Wireless Access Points), 413

catalogs, Puppet, 607

Catalyst 9800 Series wireless controllers, 417–418
- cloud-managed deployments, 419
- embedded controllers, for switches, 419–420

CBWFQ (Class-Based Weighted Fair Queuing), 343–344

CEF (Cisco Express Forwarding), 22, 24–26
- centralized switching, 26–27
- dCEF, 26–27
- polarization, 21

central switching, 416

centralized switching, 26–27

centralized web authentication, 459

centralized WLC deployments, 408
- CAPWAP, 413
- split MAC, 411–413
- traffic flows, 413–415

centrally switched mode, AP, 426

certificate management, SD-WAN, 591

Chef, 610
- clients, 611
- concepts, 610–611
- configuration management, 600–601
- configuration script, 610
- cookbooks, 611
- examples, 611–612
- nodes, 611
- recipes, 611
- resources, 611

- servers, 610
- system management, 610–612

Cisco AMP, endpoint security, 473–475

Cisco AnyConnect, endpoint security, 472–473

Cisco AutoQoS, 347

Cisco Catalyst 9800 Series wireless controllers
- cloud-managed deployments, 419
- embedded controllers, for switches, 419–420

Cisco CMX, 442–443
- analytics tools, 444
    - *location analytics, 444*
    - *presence analytics, 444*
- Connect feature, 443
- Detect feature, 443
- Engage feature, 443

Cisco Collective Security Intelligence Cloud, 475

Cisco DNA Center, 533, 535–536
- analytics, 536
- application policies, 347–348
- assurance, 541–542, 548–550
    - *AI-Driven Data, 550–551*
    - *Application Health, 548–550*
    - *Baseline Deviation Reporting, 551*
    - *Client 360, 551–552*
- authentication, 589
- automation, 536, 542–548
- cloud service management, 536
- consumers, 347–348
- digital transformation, need for, 533–535
- features, 539–541
- IBN, 537–539
- identity management, 536
- infrastructure function, 536
- integration API, 590
- Intent API, 585–586, 588
- inventory management, 543–544
- IP pools, 545
- ISE, 276
- IT4IT reference architecture, 590
- Know Your Network REST request paths, 588
- multivendor support, 589–590
- Network Plug and Play, 546–547
- networks
    - *discovery process, 542–543*
    - *hierarchies, 546*
    - *managing, 542–543*
    - *provisioning, 546–547*
- producers, 347
- REST API, 587–590
- SD-Access, 270–272
- security, 536–537
- software image management, 544–545
- Template Editor, 547–548
- virtualization, 536

Cisco EasyConnect, 490

Cisco ENCS, 526–527

Cisco Enterprise NFV, 523
- architectures, 523–524
- NFVIS, 524–527

Cisco E-Series servers, 526

Cisco FlexConnect, deployments, 408, 415–416

Cisco IOS CLI, EEM policies, 618–619

Cisco IOS IP SLA, 360–365

Cisco IOX FlexVPN, 235

Cisco ISR 4000 routers, 526

Cisco Meraki, cloud-managed deployments, 408, 416–417

Cisco Mobility Express, 422

Cisco SD-Access, 11–12, 269–270
- Cisco DNA Center, 270–272
- fabric technology, 269, 272–274
    - *components, 277–279*
    - *control plane, LISP, 275*
    - *data plane, LISP, 275*
    - *policy plane, 275–276*
- fusion routers, 282–283
- overview, 272–274
- shared services, 281–283
- topologies, 282–283

Cisco Supervisor Engine
- hardware redundancy, 28–29
- RPR, 28
- RPR+, 28–29
- SSO, 29

Cisco Talos website, 468

Cisco UCS C-Series servers, 526

Cisco vManage, 288–289, 297
- GUI, 289
- high resiliency mode, 290
- multitenant mode, 290
- NMS, 288
- Postman, REST API operations, 596–598

Python, REST API operations, 594–596
RBAC, 289
REST API, 590–598

class maps, 514

classes, Puppet, 607

classification tools, QoS, 329–332
802.11 Wireless QoS: 802.11e, 336–337
802.1p Class of Service, 334–336

CLI event detectors, 616

Client 360, DNA Center Assurance, 551–552

client authentication, wireless
802.1X user authentication, 449–451, 453–457
EAP, 452–457
PSK authentication, 446–449
WebAuth, guest access with, 457–458
auto-anchor mobility (guest tunneling), 458
centralized web authentication, 459
configuring WebAuth, 459–463
local web authentication, 458
local web portal, external authentication, 459
WPA, 445–449

Client Connector, 475

client mode (VTP), 47

clients, Chef, 611

clock offset, NTP, 219

Clos architectures, 12–13

cloud computing
Cisco Collective Security Intelligence Cloud, 475
cloud-managed deployments
Catalyst 9800 Series wireless controllers, 419
Meraki, 408, 416–417
cloud service management, DNA Center, 536
WAN Edge cloud routers, 287

cloud recall (retrospective alerting), 475

CMX (Connected Mobile Experiences), 442–443
analytics tools, 444
location analytics, 444
presence analytics, 444
Connect feature, 443
Detect feature, 443
Engage feature, 443

collapsed core (two-tier) design, 4–5

color, SD-WAN, 297–298

Command Runner API, 589

comments, idiomatic Python (being Pythonic), 558–559

community attribute, BGP, 176

concatenating strings, Python, 560

conditional debug command, 358–359

confidentiality, IPsec, 236

configuration, SD-WAN, 591

Configuration BPDU, 57

configuration datastores, NETCONF, 580

configuration management tools (DevOps), 599–601
Chef, 600–601
Puppet, 600
SaltStack, 601

configuring
802.1Q trunk ports, 41
AAA framework
accounting, 503
authorization, 501–503
access, AAA framework, granular access control, 496
ACL
named ACL, 510–511
numbered ACL, 508–510
agent-based configuration, 599
agentless configuration, 599
basic settings, Review Lab 1, 624
BGP, 176–182
BPDU Guard, 78–79
Chef configuration script, 611–612
conditional debug command, 358
CoPP, 514–515
DTP, 45–46
dynamic NAT, 216
EBGP, 176–178
ERSPAN, 371
EtherChannel, 95–96
Flexible NetFlow, 385–386
GRE
example, 229–230
steps, 228–229
HSRP, 197–199
IOS IP SLA, 363–364
LACP, 97–98
fast rate timer, 103–104

    - *max bundle feature, 102–103*
    - *port channel min-links, 103*
    - *port priority, 103*
    - *system priority, 103*
  - local SPAN, 366–367
  - local users, AAA framework, 499–500
  - MHSRP, 195
  - MST, 87–89
  - multiarea OSPF, 137–138
  - named ACL, 510–511
  - NAT, 214
    - *dynamic NAT, 214, 216*
    - *PAT, 215–217*
    - *static NAT, 214–215*
  - NBAR, 332
  - NetFlow, 382
  - NTP, 223–224
  - numbered ACL, 508–510
  - object tracking, HSRP, 193–194
  - OSPF, multiarea OSPF, 137–138
  - OSPFv3, 157–160
  - PAgP, 98–99
  - passwords, device access security, 494–495
  - PAT, 215–217
  - proxy-agent configuration, 599
  - Puppet configuration script, 608–610
  - RADIUS, 500–501
  - remote SPAN, 369
  - Root Guard, 80
  - routed subinterfaces, 51–52
  - routed switch ports, 54
  - routing protocols, Review Lab 1, 625–627
  - RSTP, 68–69, 76–77
  - SaltStack configuration script, 614–615
  - static NAT, 214–215
  - static routing, Review Lab 2, 636–637
  - STP Loop Guard, 80–81
  - STP PortFast, 78–79
  - SVI, 53
  - TACACS+, 501
  - UDLD, 82–83
  - usernames, device access security, 494–495
  - VRF, Review Lab 2, 636–637
  - VRF-Lite, 530–531
  - VRRP, 201–203
  - VTP
    - *configuration example, 49*
    - *configuration revision, 48*
  - WebAuth, 459–463

- **congestion management, QoS, 330, 342**
  - congestion avoidance, 344–345
  - scheduling tools, 342
- **Connect feature, CMX, 443**
- **Connect neighbor state, BGP, 171**
- **connected mode, AP, 426–427**
- **connectivity testing, ping command, 353**
- **consoles, syslog messages, 374**
- **consumers, 347–348**
- **content security, 470–471**
- **context awareness, firewalls, 478**
- **control plane**
  - API, 583
  - data path virtualization, 528–529
  - LISP, 258
  - packet forwarding, 19
  - SD-WAN, 288, 290–291
- **control plane node, fabric devices, 277–278**
- **control station, Ansible, 602**
- **controllers**
  - fabric technology, 277, 279
  - SDN controllers, API, 584
  - vSmart controllers, 288–291, 297–298, 301
  - WLC
    - *AP discovery operation, 422–425*
    - *AP modes, 425–428*
    - *Catalyst 9800 Series wireless controllers, 417–420*
    - *central switching, 416*
    - *centralized deployments, 408, 410–415*
    - *Cisco Mobility Express, 422*
    - *fabric technology, 277, 279*
    - *local switching, 416*
    - *split tunneling, 416*
    - *WebAuth, 489*
- **converged networks**
  - bandwidth, 324
  - delay, 324–325
    - *processing delay (variable), 325*
    - *propagation delay (fixed), 326*
    - *queuing delay (variable), 325*
    - *serialization delay (fixed), 325*
  - jitter, 325, 326
    - *dejitter buffer operation, 326–327*
    - *encapsulation, 326*
    - *fragmentation, 326*

- network latency, 324–326
- packet loss, 325, 327
- QoS, 324–327

convergence
- rapid convergence, EIGRP, 106
- RSTP, 67

cookbooks, Chef, 611

CoPP (Control Plane Policing), 513–514
- class maps, 514
- configuring, 514–515
- policy maps, 514
- service policies, 514

core layer (hierarchical LAN design), 2, 4

CQ (Custom Queuing), 343

CRC (Cyclic Redundancy Checks), 20

CRUD operations, 585–586

crypto maps, 243–245

D

data (forwarding) plane, API, 583

data analysis, NetFlow, 380–381

data encoding, 567
- JSON, 567–570
- XML, 570
    - *namespaces, 571–572*
    - *NETCONF, XML encoding, 579*
    - *Python, XML example, 570–571*

data integrity, IPsec, 237

data loss and exfiltration, vectors of, 467

data models, network programmability, 572–573
- NETCONF, 575–577
    - *configuration datastores, 580*
    - *southbound API, 584*
    - *SSH, 577–579*
- REST, 574–575
- RESTCONF, 580
    - *GET example, 581–582*
    - *methods, 580–581*
- schema trees, 573
- YANG, 573–574

data path virtualization, 527, 528

data plane
- LISP, 255–258
- packet forwarding, 19
- SD-WAN, 291–292

data retry counts, wireless roaming, 434

data types, Python
- Boolean data type, 559, 562
- dictionary data type, 559
- file data type, 559
- list data type, 559
- numbers data type, 559, 561–562
- overview, 559
- string data type, 559–560
    - *built-in methods, 561*
    - *concatenating strings, 560*
    - *printing strings, 560*

databases
- LSDB, building, 127
- MAC databases, instability, 56
- topological (link-state) databases, 122

Day 0 network provisioning, 546–547

Day N network automation, 547–548

DBD packets, OSPF, 129

dCEF (Distributed Switching), 26–27

DDoS attacks, 466

dead intervals, hello packets, OSPF, 126

debug command, 357
- access lists, 359–360
- conditional debug command, 358–359

decibels (dB), 395–397

default gateways
- failovers, FHRP, 185
- redundancy, 183–184

DEI (Drop Eligible Indicators), 40

dejitter buffer operation, 326–327

delay, 325
- converged networks, 324–325
    - *processing delay (variable), 325*
    - *propagation delay (fixed), 326*
    - *queuing delay (variable), 325*
    - *serialization delay (fixed), 325*
- end-to-end delay (network latency), 324–326

Delay (K3) metric, EIGRP, 110

DELETE action, REST API, 585–586

DELETE method, RESTCONF, 581

DES algorithm, IPsec, 237

describe conditionals, Python, 562–563

designated ports, 62–64

Desirable mode (PAgP), 95

Detect feature, CMX, 443

device access security, 492
  enable secret password, 492–493
  line passwords, 493–494
  passwords, configuring, 494–495
  usernames, 494–495
device inventories, SD-WAN, 591
device management, SNMP, 565–567
device registration WebAuth, 490
device virtualization, 527
DevOps (configuration management tools), 599–601
  Chef, 600–601
  Puppet, 600
  Salt, 601
DH (Diffie-Hellman) key agreements, 237–238
DHCP, WLC discovery process, 423
diagnostics, troubleshooting principles, 350–351
dictionary data type, Python, 559
DiffServ (Differentiated Services) model, 328–329
digital transformation, need for, 533–535
direct topology changes, 58–59
directed probes, wireless roaming, 435
directional antennas, 429
disabled ports, 64
disabled state (ports), 63, 64
discarding state (ports), 65
discovery process
  DNA Center, 542–543
  WLC, 422
distribute lists, OSPF route filtering, 151–152, 154
distribution layer (hierarchical LAN design), 2, 3
distribution trees, multicasting, 313
  shared trees, 314–315
  source trees, 313, 315
DMVPN (Dynamic Multipoint VPN), 234
DMZ (Demilitarized Zones), 477
DNA Center, 533, 535–536
  analytics, 536
  application policies, 347–348
  assurance, 541–542, 548–550
    AI-Driven Data, 550–551
    Application Health, 548–550
    Baseline Deviation Reporting, 551
    Client 360, 551–552
  authentication, 589
  automation, 536, 542–548
  cloud service management, 536
  consumers, 347–348
  digital transformation, need for, 533–535
  features, 539–541
  IBN, 537–539
  identity management, 536
  infrastructure function, 536
  integration API, 590
  Intent API, 585–586, 588
  inventory management, 543–544
  IP pools, 545
  ISE, 276
  IT4IT reference architecture, 590
  Know Your Network REST request paths, 588
  multivendor support, 589–590
  network
  Network Plug and Play, 546–547
  networks
    discovery process, 542–543
    hierarchies, 546
    managing, 542–543
    provisioning, 546–547
  producers, 347
  REST API, 587–590
  SD-Access, 270–272
  security, 536–537
  software image management, 544–545
  Template Editor, 547–548
  virtualization, 536
DNS (Domain Name Service), WLC discovery process, 423
documentation, SD-WAN API libraries, 592–598
DoS attacks, 466
Down state, OSPF neighbors, 128
DR (Designated Routers)
  hello packets, 126
  OSPF DR election process, 134–136
DSCP, per-hop behaviors, 338–339
DTP (Dynamic Trunking Protocol), 44–45
  combinations, 45
  configuring, 45–46
  verifying status, 44–46

dual multihomed connections, BGP, 168

dual-homed connections, BGP, 168

dwell time, 443, 444

dynamic auto mode, switchport mode dynamic auto command, 44

dynamic desirable mode, switchport mode dynamic desirable command, 44

Dynamic Interpreter. *See* Python, Python Shell

dynamic NAT, 208–211, 214, 216

E

EAP (Extensible Authentication Protocol), 452–457

EasyConnect, 490

EBGP (External Border Gateway Protocol), 172–173
- configuring, 176–182
- verifying, 178–182

ECDH (Elliptic Curve Diffie-Hellman) key agreements, 238

ECDSA signatures, 239

ECMP (Equal-Cost Multipathing), 106

edge node, fabric devices, 277

edge ports, 66

EEM (Embedded Event Manager), 615
- applets, 616–619
- architectures, 615
- Cisco IOS CLI, writing EEM policies, 618–619
- event detectors, 615–617
- policies, 615–620
- servers, 615–616
- tcl scripts, 616, 618–620

EGP (Exterior Gateway Protocol), BGP, 164

EID (Endpoint Identifiers)
- addresses, 252–253, 255
- subnets, border nodes, 278

EIGRP (Enhanced Interior Gateway Routing Protocol), 105–106
- AS numbers, 109
- authentication, 110
- bandwidth, reduced use, 106
- ECMP, 106
- encapsulation, 106–107
- FD, 116
- feasible successors, 107
- K values, 109
  - *Bandwidth (K1) metric, 110*
  - *calculating, 110–111*
  - *Delay (K3) metric, 110*
  - *Load (K2) metric, 110*
  - *Reliability (K4, K5) metric, 110*
  - *verifying, 111–112*
  - *wide metrics, 112–115*
- load balancing, 106, 117–118
  - *equal-cost load balancing, 118*
  - *unequal-cost load balancing, 118–119*
- loop-free, classless routing protocol, 106
- loop-free path selection, 116–117
- metrics (K values), 109
  - *Bandwidth (K1) metric, 110*
  - *calculating, 110–111*
  - *Delay (K3) metric, 110*
  - *Load (K2) metric, 110*
  - *Reliability (K4, K5) metric, 110*
  - *verifying, 111–112*
  - *wide metrics, 112–115*
- multi-address family support, 106
- neighbor adjacencies, 109–110
- operation overview, 107–108
- packets, 108–109
- passwords, 110
- path selection, 115–117
- rapid convergence, 106
- RD, 116
- reliable transport protocol, 106–107
- subnets, 109
- successor routes, 107

EIRP (Effective Isotropic-Radiated Power), 398–399

elections
- designated port elections, 62–63
- root bridges, 59–60
- root ports, 60–63

embedded controllers, Catalyst 9800 Series wireless controllers, 419–420

enable secret password, 492–493

encapsulation
- EIGRP, 106–107
- encapsulation dot1q command, 51–52

ERSPAN, 370–372
ESP, 236
GRE, 227–228
configuring, 228–230
verifying, 230–232
jitter, 326
OSPF packets, 129
switchport trunk encapsulation dot1q command, 41
VXLAN, 263–265, 278

encoding data, 567
JSON, 567–570
XML, 570
namespaces, 571–572
NETCONF, XML encoding, 579
Python, XML example, 570–571

encryption
RSA encrypted nonces, 239
symmetric encryption algorithms, 237–238

ENCS (Enterprise Network Compute System), 526–527

endpoints
authentication, 802.1X, 485–486
EID
addresses, 252–253, 255
subnets, border nodes, 278
registration, 278
security, 471–475

end-to-end delay (network latency), 324–326

end-to-end VLAN, 34

Engage feature, CMX, 443

enhanced object tracking event detectors, 616–617

enterprise networks
architectures, 4
Clos architectures, 12–13
Layer 2 access layer (STP-based) design, 6–7
Layer 2 looped design, 10
Layer 2 loop-free design, 10
Layer 3 access layer (router-based) design, 7–8, 11
simplified campus design, 9–11
software-defined access (SD-access) design, 11–12
spine and leaf architectures, 12–13
StackWise, simplified campus design, 9–10
three-tier design, 5–6
two-tier (collapsed core) design, 4–5
VSS, simplified campus design, 9–11
heirarchical LAN design, 1–2
access layer, 2–3
core layer, 2, 4
distribution layer, 2, 3

Enterprise NFV, 523
architectures, 523–524
NFVIS, 524–527

entry points, Python scripts, 564–565

E-plane, omnidirectional antennas, 429

equal-cost load balancing, EIGRP, 118

ERSPAN (Encapsulated Remote SPAN), 370–372

ESP (Encapsulating Security Payloads), 236

Established neighbor state, BGP, 171

EtherChannel, 91
configuring, 95–96
LACP, 93–94
configuring, 97–98
fast rate timer, 103–104
hot-standby ports, 101–102
max bundle feature, 102–103
port channel min-links, 103
port priority, 103
system priority, 103
load balancing, 96–97
need for, 91–93
PAgP, 93–95, 98–99
verifying, 97, 99–101

ETR (Egress Tunnel Routers), 254, 255

event detectors, EEM, 615–617

Exchange state, OSPF neighbors, 128

exfiltration, vectors of data loss and, 467

exit points, fabric domains, 278

exploits, network security, 466

export data format, NetFlow, 381–382

ExStart state, OSPF neighbors, 128

extended ACL, 508, 513

extended ping command, 353–355

extended subnets, host mobility, 259–260

Extended System ID, 60

external authentication, local web portal, 459

external routers, inter-VLAN routing, 51–52

F

fabric technology, 269
  AP, 277, 279–281
  border node, 277, 278
  control plane node, 277–278
  edge node, fabric devices, 277
  intermediate node, 277, 279
  overlay networks, 274
  SD-Access, 272–274
    components, 277–279
    control plane, LISP, 275
    data plane, LISP, 275
    policy plane, 275–276
  underlay networks, 274
  wireless controllers, 277, 279

failovers
  AP, 423–424
    AP Fallback, 424–425
    prioritizing, 424
  FHRP
    default gateway failovers, 185
    failover process, 185–186

Fallback, AP, 424–425

fast rate timer, LACP, 103–104

fast switching, 20–23

FD (Feasible Distance), EIGRP, 116

feasible successors, EIGRP, 107

FHRP (First Hop Redundancy Protocol)
  default gateway failovers, 185
  failover process, 185–186
  GLBP, 185
  HSRP, 185, 186
    active routes, 187
    authentication, 195–196
    configuring, 197–199
    forwarding, 188
    groups, 187–189
    load balancing, 194
    MHSRP, 194–195
    multigroups, 194–195
    object tracking, 191–194
    preempting, 189–190
    prioritizing, 189
    standby groups, 186–187
    standby routes, 187
    states, 191
    timers, 190–191
    versions, 196–197
  operations, 184–185
  VRRP, 185, 199–200
    authentication, 201
    configuring, 201–203

FIB (Forwarding Information Base) tables
  CEF, 21, 24
  NSF, 29

FIFO (First-In, First-Out), 343

File API, 589

file data type, Python, 559

filtering, URL, 478

firewalls, 472, 476–477

first-hop redundancy, configuring, Review Lab 1, 627–630

Flash memory, syslog messages, 375

FlexConnect
  deployments, 408, 415–416
  FlexConnect mode, AP, 425–427

flexibility, AAA framework, 496

Flexible NetFlow, 384
  configuring, 385–386
  NetFlow versus, 384–385
  verifying, 386–387

FlexVPN, 235

flow analyzers, 378

flow collectors, 378

flow exporters, 378

formulas (states), SaltStack, 613

forwarding
  LISP, 278
  multicasting, 305
  packets, 19
    control plane, 19
    data plane, 19
    Layer 2 forwarding tables, 18
    Layer 3 forwarding tables, 18
    NSF, 29–30

forwarding plane
  API, 583
  data path virtualization, 529

forwarding state (ports), 63–65

fragmentation, jitter, 326

free path loss, RF communications, 392–393

frequencies
  OFDM, 400
  RF communications, 390

Full state, OSPF neighbors, 129

fully meshed network topologies, 233

fusion routers, 282–283

G

gateways
  Anycast Layer 3 gateways, 278
  VXLAN, 265–266

GET action, REST API, 585, 586

GET example, RESTCONF, 581–582

GET method, RESTCONF, 580

GLBP (Gateway Load Balancing Protocol), 185

GLOP blocks (233.0.0.0/8), 309

grains, SaltStack, 613

GRE (Generic Route Encapsulation), 227–228
  configuring
    example, 229–230
    steps, 228–229
  GRE over IPsec
    crypto maps, 243–245
    site-to-site VPN, 243–244
    transport mode, 242
    Tunnel IPsec Profiles, 245–246
    tunnel mode, 242–243
  verifying, 230–232

group addresses, multicasting, 307–309

group-specific queries, IGMPv2, 312

guest access with WebAuth, 457–458
  auto-anchor mobility (guest tunneling), 458
  centralized web authentication, 459
  configuring WebAuth, 459–463
  local web authentication, 458
  local web portal, external authentication, 459

guest tunneling (auto-anchor mobility), 441–442, 458

GUI (Graphical User Interface), vManage, 289

H

hacking tools, 468

hardware redundancy
  Cisco Supervisor Engine, 28–29
  RPR, 28
  RPR+, 28–29
  SSO, 29

heirarchical LAN design, 1–2
  access layer, 2–3
    Layer 2 access layer (STP-based) design, 6–7
    Layer 2 looped design, 10
    Layer 2 loop-free design, 10
    Layer 3 access layer (router-based) design, 7–8, 11
    simplified campus design, 9–10
    software-defined access (SD-access) design, 11–12
  core layer, 2, 4
  distribution layer, 2, 3

hello intervals, hello packets, 126

hello packets, 108
  Area ID, 126
  authentication, 126
  BDR, 126
  dead intervals, 126
  DR, 126
  hello intervals, 126
  Neighbors field, 126
  OSPF, 125–126, 129
  router ID, 126
  router priority, 126
  stub area flags, 126

helper utilities, Python, 557–558

hierarchies of networks, DNA Center, 546

high availability, 55

high resiliency mode, vManage, 290

HMAC (Hash-Based Message Authentication Codes), 238

hop-by-hop path selection, BGP, 166–167

host mobility, LISP, 259
  across subnets, 260–261
  example of, 260–262
  extended subnets, 259–260

host support, Review Lab 1, 624–625

hosted hypervisors, 521–522

hot-standby ports, LACP, 101–102

H-plane, omnidirectional antennas, 429

HSRP (Hot Standby Routing Protocol), 185, 186
- active routes, 187
- authentication, 195–196
- configuring, 197–199
- forwarding, 188
- groups, 187–189
- load balancing, MHSRP, 194
- multigroups, 194–195
- object tracking, 191–194
- preempting, 189–190
- prioritizing, 189
- standby groups, 186–187
- standby routes, 187
- states, 191
- timers, 190–191
- versions, 196–197

HTTP (HyperText Transfer Protocol), status codes, 586

hub-and-spoke network topologies, VPN, 233

hypervisors, 520–522

I

IaaS (Infrastructure as a Service), 295

IaC (Infrastructure as Code), 600

IBGP (Internal Border Gateway Protocol), 173

IBN (Intent-Based Networking), 537–539

IBNS (Identity-Based Networking Services), 484–485
- 802.1X endpoint authentication, 485–486
- authorization, 487
- MAB, 487–488
- message flow, 486–487
- WebAuth, 489–490

identification
- Area ID, hello packets, 126
- BID, 59, 60
- bridge ID, verifying, RSTP, 69–70
- Extended System ID, root bridges, 60
- LISP, 251, 252
  - *ALT routers, 255*
  - *components, 253*
  - *control plane, 258, 275*
  - *data plane, 255–258, 275*
  - *EID addresses, 252–253, 255*
  - *ETR, 254*
  - *forwarding, 278*
  - *host mobility, 259–262*
  - *IP routing versus, 252*
  - *ITR, 253–254*
  - *mapping LISP instances to VRF instances, 278*
  - *MR, 254*
  - *MS, 254*
  - *PETR, 255*
  - *PITR, 255*
  - *RLOC addresses, 252–253*
- process ID, OSPF, 123–124
- router ID, hello packets, 126
- router ID, OSPF, 124–125
- site ID, SD-WAN, 296

identity management
- DNA Center, 536
- IBNS, 484–485
  - *802.1X endpoint authentication, 485–486*
  - *authorization, 487*
  - *MAB, 487–488*
  - *message flow, 486–487*
  - *WebAuth, 489–490*

idiomatic Python (being Pythonic), 558–559

Idle neighbor state, BGP, 171

IEEE wireless standards, 399
- 802.11a wireless standard, 400
- 802.11ac wireless standard, 400
- 802.11ax (Wi-Fi 6) wireless standard, 401
- 802.11b wireless standard, 399–400
- 802.11g wireless standard, 400
- 802.11n wireless standard, 400

IGMP (Internet Group Management Protocol)
- architectures, 311
- IGMPv1, 311–312
- IGMPv2, 312
- IGMPv3, 312–313

IGP (Interior Gateway Protocol), BGP, 164

IKE (Internet Key Exchange), 236, 239–240
- IKEv1, 240–241
- IKEv2, 242

image management, software, 544–545
inbound ACL, 511–512
indentation (spaces/tabs), idiomatic Python (being Pythonic), 559
indirect topology changes, 59
individual point-to-point VPN connections, 232–233
infrastructure function, DNA Center, 536
infrastructure security
  AAA framework, 497
    accounting, 495, 503
    authentication, 495, 496, 498–499
    authorization, 495, 501–503
    configuring local users, 499–500
    enabling, 499–500
    flexibility, 496
    granular access control, 496
    overview, 495–496
    RADIUS, 496–498, 500–501
    redundancy, 496
    scalability, 496
    TACACS+, 497, 498, 501
  ACL, 504
    applying to interfaces, 511–513
    extended ACL, 508, 513
    inbound ACL, 511–512
    named ACL, 508, 510–511
    numbered ACL, 508–510
    outbound ACL, 511
    overview, 504–505
    standard ACL, 507–508, 512
    wildcard masking, 505–507
  CoPP, 513–514
    class maps, 514
    configuring, 514–515
    policy maps, 514
    service policies, 514
  device access security, 492
    enable secret password, 492–493
    line passwords, 493–494
    passwords, 494–495
    usernames, 494–495
  secure access control, 491
Init state, OSPF neighbors, 128
Initial state, HSRP, 191
inside global addresses, NAT, 207
inside local addresses, NAT, 207
insignificant topology changes, 59
instability, MAC databases, redundant topologies, 56
instances, MST, 85–86
integration API, 590
integrity of data, IPsec, 237
Intent API, Data Center, 585–586, 588
inter-controller traffic flows, centralized WLC deployments, 414
interdomain routing, BGP, 163–164
interface addressing, Review Lab 1, 624
interface event detectors, 616
intermediate node, fabric devices, 277, 279
internal routers, OSPF, 133
Internetwork Control blocks (224.0.1/24), multicast addressing, 309
inter-VLAN routing, 34, 50–51
  encapsulation dot1q command, 51–52
  external routers, 51–52
  subinterfaces, 51–52
  SVI, 52–53
intra-area prefix LSA, 157
intra-controller traffic flows, centralized WLC deployments, 414–415
IntServ (Integrated Services) model, 328
inventories
  device inventories, SD-WAN, 591
  managing, DNA Center, 543–544
inventory files, Ansible, 602, 605–606
IOS CLI, EEM policies, 618–619
IOS FlexVPN, 235
IOS IP SLA, 360–365
IP (Internet Protocols)
  pools, DNA Center, 545
  private IP addresses, SD-WAN, 297
  public IP addresses, SD-WAN, 297
  routing
    LISP versus, 252
  routing tables, BGP, 169
  system IP addresses, SD-WAN, 296
IP multicasting, 303
  addressing
    Ad hoc blocks, 309
    Administratively Scoped blocks (239.0.0.0/8), 309
    GLOP blocks (233.0.0.0/8), 309

    group addresses, 307–309
    Internetwork Control blocks (224.0.1/24), 309
    Local Network Control blocks (224.0.0/24), 308
    SDP/SAP blocks (224.2.0.0/16), 309
    Source-Specific Multicast blocks (232.0.0.0/8), 309
  applications, 306–307
  bandwidth, 306
  benefits, 306
  communication, 303
  distribution trees, 313
    shared trees, 314–315
    source trees, 313, 315
  drawbacks, 306
  forwarding, 305
  IGMP
    architectures, 311
    IGMPv1, 311–312
    IGMPv2, 312
    IGMPv3, 312–313
  operations, 305
  overview, 303
  PIM, 316
    BIDIR-PIM mode, 317
    PIM-DM, 316–317
    PIM-SM, 317–319
    RP, 318, 319–321
    SSM, 317
  routing, 315–316
  service model, 310
  SLA
    event detectors, 616
    functionality, Review Lab 1, 627–630
  traffic streams, 304–305
  UDP, 306
  unicasting versus, 304–305

IPS (Intrusion Prevention Systems), 468–469, 478

IPsec (Internet Protocol Security), 227
  3DES algorithm, 237
  AES, 237–238
  AH, 236
  antireplay protection, 237
  confidentiality, 236
  data integrity, 237
  DES algorithm, 237
  ECDSA signatures, 239
  ESP, 236
  GRE over IPsec
    crypto maps, 243–245
    site-to-site VPN, 243–244
    Tunnel IPsec Profiles, 245–246
  GRE over IPsec transport mode, 242
  HMAC, 238
  IKE, 236, 239–240
    IKEv1, 240–241
    IKEv2, 242
  IPsec over GRE tunnel mode, 242–243
  key management, 237
  origin authentication, 237
  PSK, 238
  RSA, 237
    encrypted nonces, 239
    signatures, 238
  SA, 239–240
  services, 236–239
  VPN, 238–239
    GRE over IPsec transport mode, 242
    IPsec over GRE tunnel mode, 242–243
    overview, 235–236
    site-to-site VPN, 232–235, 242–249
    transport mode, 235–236
    tunnel mode, 236
  VTI, 243
  VTI over IPsec, 247–249

IPv4 addressing
  addressing tables
    Review Lab 1, 623
    Review Lab 2, 634–635
  multicasting
    Ad hoc blocks, 309
    Administratively Scoped blocks (239.0.0.0/8), 309
    GLOP blocks (233.0.0.0/8), 309
    group addresses, 307–309
    Internetwork Control blocks (224.0.1/24), 309
    Local Network Control blocks (224.0.0/24), 308
    SDP/SAP blocks (224.2.0.0/16), 309
    Source-Specific Multicast blocks (232.0.0.0/8), 309
  OSPFv3, 161–162
  WLC discovery process, 422

IPv6 addressing
  addressing tables
    Review Lab 1, 623
    Review Lab 2, 634–635

  - OSPFv3, 161–162
  - WLC discovery process, 422

ISAKMP (Internet Security Association and Key Management Protocol)
  - crypto maps, 244–245
  - GRE over IPsec, Tunnel IPsec Profiles, 245–246
  - IKE, 239–241
  - site-to-site VTI over IPsec, 247, 249

ISE (Identity Services Engine), Cisco DNA Center, 276

isotropic antennas, 397–399

IT4IT reference architecture, 590

ITR (Ingress Tunnel Routers), 253–255

ITSM (IT Service Management), 590

J

jitter, 325, 326
  - dejitter buffer operation, 326–327
  - encapsulation, 326
  - fragmentation, 326
  - IOS IP SLA, 363–364

join messages, IGMPv3, 312–313

join orders, AP, 423

joined hub-and-spoke topologies, 233

JSON (JavaScript Object Notation), 567–570

K

K values, 109
  - Bandwidth (K1) metric, 110
  - calculating, 110–111
  - Delay (K3) metric, 110
  - Load (K2) metric, 110
  - Reliability (K4, K5) metric, 110
  - verifying, 111–112
  - wide metrics, 112–115

KEEPALIVE messages, BGP, 164, 170

key management
  - DH key agreements, 237–238
  - ECDH key agreements, 238
  - IKE
    - *IKEv1, 240–241*
    - *IKEv2, 242*
  - IPsec, 237
  - ISAKMP
    - *crypto maps, 244–245*
    - *GRE over IPsec, Tunnel IPsec Profiles, 245–246*
    - *IKE, 239–241*
    - *site-to-site VTI over IPsec, 247, 249*
  - PSK authentication, 446–449

Know Your Network REST request paths, 588

L

LACP (Link Aggregation Control Protocol), 93–94
  - active mode, 94
  - configuring, 97–98
  - fast rate timer, 103–104
  - hot-standby ports, 101–102
  - max bundle feature, 102–103
  - passive mode, 94
  - port channel min-links, 103
  - port priority, 103
  - system priority, 103

LAN (Local Area Networks). ***See also*** **VLAN**
  - hierarchical design, 1–2
    - *access layer, 2–3, 6–12*
    - *core layer, 2, 4*
    - *distribution layer, 2, 3*
  - VXLAN, 251, 262–263
    - *encapsulation, 263–265*
    - *encapsulation/decapsulation, 278*
    - *gateways, 251*
    - *VXLAN-GPO headers, 266–267*
  - WLAN
    - *autonomous AP deployments, 408–410*
    - *centralized WLC deployments, 408, 410–415*
    - *cloud-managed deployments, Meraki, 408, 416–417*
    - *FlexConnect deployments, 408, 415–416*
    - *SD-Access deployments, 408*
  - WLC
    - *AP discovery operation, 422–425*
    - *AP modes, 425–428*
    - *Catalyst 9800 Series wireless controllers, 417–420*
    - *central switching, 416*
    - *centralized deployments, 408, 410–415*
    - *Cisco Mobility Express, 422*
    - *local switching, 416*
    - *split tunneling, 416*

latency, network, 324–326

Layer 2
- access layer (STP-based) design, 6–7
- classification tools, QoS, 334
  - *802.1p Class of Service, 334–336*
  - *802.11 Wireless QoS: 802.11e, 336–337*
- EtherChannel, LACP, configuring, 97–98
- forwarding tables, 16, 18
- looped design, 10
- loop-free design, 10
- marking tools, QoS, 334
  - *802.1p Class of Service, 334–336*
  - *802.11 Wireless QoS: 802.11e, 336–337*
- networks, configuring
  - *Review Lab 1, 624–625*
  - *Review Lab 2, 637*
- redundancy, 55
- roaming, 437–439
- switches
  - *ACL, 17*
  - *CAM tables, 15–17*
  - *Layer 2 forwarding tables, 16*
  - *MAC address tables, 17*
  - *QoS, 17*
  - *TCAM tables, 17*
  - *traffic switching process, 16–17*
  - *unknown unicast flooding, 15*
- virtualization, 527–529

Layer 3
- access layer (router-based) design, 7–8, 11
- EtherChannel, PAgP, configuring, 98–99
- forwarding tables, 18
- marking tools, QoS
  - *DSCP per-hop behaviors, 338–339*
  - *mapping Layer 2 to Layer 3 markings, 339–340*
  - *Type of Service, 337–338*
- roaming, 439–441
- switches
  - *ACL, 18*
  - *CAM tables, 18, 19*
  - *QoS, 18*
  - *TCAM tables, 18*
  - *traffic switching process, 17–19*
- virtualization, 527–529

learning state (ports), 63, 65

leave-group messages, IGMPv2, 312

line passwords, 493–494

link LSA, 157

link type, RSTP, rapid transition to forwarding state, 66

links, UDLD, 82–83

link-specific tools, QoS, 330

link-state (topological) databases, 122

LISP (Locator/ID Separation Protocol), 251, 252
- ALT routers, 255
- components, 253
- control plane, 258
- data plane, 255–258
- EID addresses, 252–253, 255
- ETR, 254
- forwarding, 278
- host mobility, 259
  - *across subnets, 260–261*
  - *example of, 260–262*
  - *extended subnets, 259–260*
- IP routing versus, 252
- ITR, 253–254
- mapping LISP instances to VRF instances, 278
- MR, 254
- MS, 254
- PETR, 255
- PITR, 255
- RLOC addresses, 252–253

list data type, Python, 559

Listen state, HSRP, 191

listening state (ports), 63

LLQ (Low-Latency Queuing), 342–344

Load (K2) metric, EIGRP, 110

load balancing
- EIGRP, 106, 117–118
  - *equal-cost load balancing, 118*
  - *unequal-cost load balancing, 118–119*
- EtherChannel, 96–97
- GLBP, 185
- HSRP, MHSRP, 194
- MHSRP, 194
- proprietary load-balancing, wireless roaming, 434–435

Loading state, OSPF neighbors, 128–129

local mode, AP, 425

Local Network Control blocks (224.0.0/24), multicast addressing, 308

local preference path attribute, BGP, 175

local SPAN, 366–368

local switching, 416

local VLAN, 34–35

local web authentication, 458

local web portal, external authentication, 459

locally switched mode, AP, 426

location analytics, 444

location services, wireless, 442
- accuracy, 444–445
- CMX, 442–443
  - *analytics tools, 444*
  - *Connect feature, 443*
  - *Detect feature, 443*
  - *Engage feature, 443*

logging services, 373–374
- Flexible NetFlow, 384
  - *configuring, 385–386*
  - *NetFlow versus, 384–385*
  - *verifying, 386–387*
- NetFlow, 378
  - *configuring, 382*
  - *data analysis, 380–381*
  - *export data format, 381–382*
  - *Flexible NetFlow, 384–387*
  - *flow analyzers, 378*
  - *flow collectors, 378*
  - *flow creation in NetFlow cache, 379–380*
  - *flow exporters, 378*
  - *process, 378–379*
  - *verifying, 382–384*
- SNMP, 376–378
- syslog, 374–375
  - *message format, 375*
  - *severity levels, 375–376*

loopback networks, 134

looped design, Layer 2, 10

loop-free, classless routing protocol, 106

loop-free design, Layer 2, 10

loop-free path selection, EIGRP, 116–117

loops
- bridging loops, STP, 55–56
- STP Loop Guard, 80–81

loss, packet, 325, 327

LSA (Link-State Advertisements), 122–123
- intra-area prefix LSA, 157
- link LSA, 157
- OSPF
  - *propagation, 131*
  - *route filtering, 153, 155*
  - *SPF, 145*
  - *types, 130*
- OSPFv3, 156–157

LSAck packets, OSPF, 129

LSDB (Link-State Databases), building, 127

LSR packets, OSPF, 129

LSU packets, OSPF, 129

M

MAB (MAC Authentication Bypass), 487–488

MAC addresses
- MAB, 487–488
- root bridges, 60
- tables, 17

MAC databases, instability, 56

MACsec, 483–484

malware, 467, 478

management plane, SD-WAN, 289–290

managing
- certificates, SD-WAN, 591
- cloud service management, DNA Center, 536
- configuration management tools (DevOps), 599–601
  - *Chef, 600–601*
  - *Puppet, 600*
  - *SaltStack, 601*
- congestion management, QoS, 330, 342
  - *congestion avoidance, 344–345*
  - *scheduling tools, 342*
- devices, SNMP, 565–567
- identity, DNA Center, 536
- identity, IBNS, 484–485
  - *802.1X endpoint authentication, 485–487*
  - *authorization, 487*
  - *MAB, 487–488*
  - *message flow, 486–487*
  - *WebAuth, 489–490*

- inventories, DNA Center, 543–544
- ITSM, 590
- keys
  - *DH key agreements, 237–238*
  - *ECDH key agreements, 238*
  - *IKE, 239–242*
  - *IPsec, 237*
  - *ISAKMP, 239–241, 244–247, 249*
  - *PSK, 238*
- networks
  - *DNA Center, 542–543*
  - *NMS, 288*
  - *Review Lab 1, 631–632*
  - *SNMP, 565–567*
- site management API, 588
- SNMP, 376–378
- software images, 544–545
- system management
  - *Ansible, 602–606*
  - *Chef, 610–612*
  - *EEM, 615–620*
  - *Puppet, 607–610*
  - *SaltStack, 612–615*
- VM, 522–523

manifests, Puppet, 607

man-in-the-middle attacks, 467

mapping
- class maps, 514
- crypto maps, 243–245
- Layer 2 to Layer 3 markings, 339–340
- LISP instances to VRF instances, 278
- MR, 254
- MS, 254
- policy maps, 278, 514
- RP-mapping agents, 320
- users to virtual network, 278
- wireless network markings, 340

marking tools, QoS, 329–330, 332–334
- 802.1p Class of Service, 334–336
- 802.11 Wireless QoS: 802.11e, 336–337
- Layer 2, 334–337, 339–340
- Layer 3
  - *DSCP per-hop behaviors, 338–339*
  - *mapping Layer 2 to Layer 3 markings, 339–340*
  - *Type of Service, 337–338*

master AP, Cisco Mobility Express, 422

masters, SaltStack, 612

max bundle feature, LACP, 102–103

maximal-ratio combining
- 802.11ac MIMO, 402–403
- 802.11n MIMO, 402–403

maximum data retry counts, wireless roaming, 434

MED attribute, BGP, 176

memory, syslog messages
- Flash memory, 375
- memory buffers, 374

Meraki, cloud-managed deployments, 408, 416–417

messages
- BGP
  - *KEEPALIVE messages, 170*
  - *NOTIFICATION messages, 171*
  - *OPEN messages, 169–170*
  - *UPDATE messages, 170*
- HMAC, 238
- join messages, IGMPv3, 312–313
- leave-group messages, IGMPv2, 312

metrics (K values), 109
- Bandwidth (K1) metric, 110
- calculating, 110–111
- Delay (K3) metric, 110
- Load (K2) metric, 110
- Reliability (K4, K5) metric, 110
- verifying, 111–112
- wide metrics, 112–115

MHSRP (Multiple HSRP)
- configuring, 195
- load balancing, 194

MIMO (Multiple-Input, Multiple-Output), 400
- 802.11ac MIMO, 401–402
  - *beamforming, 403–404*
  - *maximal-ratio combining, 402–403*
  - *spatial multiplexing, 402–403*
- 802.11ac MU-MIMO, 405–406
- 802.11n MIMO, 400–402
  - *beamforming, 403–404*
  - *maximal-ratio combining, 402–403*
  - *spatial multiplexing, 402–403*
- MU-MIMO, 400, 401

mines, SaltStack, 614

minions, SaltStack, 612

mobility groups, 435–436

modules
- Ansible, 602, 604
- Puppet, 607

monitor (beacons), SaltStack, 613

monitor mode, AP, 428

monitoring, SD-WAN, 591

MQC (Modular QoS CLI), 346–347

MR (Map Resolvers), 254

MS (Map Servers), 254

MST (Multiple Spanning Tree Protocol), 83–84
- configuring, 87–89
- instances, 85–86
- path costs, 89
- port priorities, 89
- regions, 84–85
- topologies, 89
- verifying, 88

multi-address family support, EIGRP, 106

multiarea OSPF, 131–132
- configuring, 137–138
- verifying functionality, 138–141

multicasting, 303
- addressing
    - *Ad hoc blocks, 309*
    - *Administratively Scoped blocks (239.0.0.0/8), 309*
    - *GLOP blocks (233.0.0.0/8), 309*
    - *group addresses, 307–309*
    - *Internetwork Control blocks (224.0.1/24), 309*
    - *Local Network Control blocks (224.0.0/24), 308*
    - *SDP/SAP blocks (224.2.0.0/16), 309*
    - *Source-Specific Multicast blocks (232.0.0.0/8), 309*
- applications, 306–307
- bandwidth, 306
- benefits, 306
- broadcast/multicast mode, NTP, 221
- communication, 303
- distribution trees, 313
    - *shared trees, 314–315*
    - *source trees, 313, 315*
- drawbacks, 306
- forwarding, 305
- IGMP
    - *architectures, 311*
    - *IGMPv1, 311–312*
    - *IGMPv2, 312*
    - *IGMPv3, 312–313*
- operations, 305
- overview, 303
- PIM, 316
    - *BIDIR-PIM mode, 317*
    - *PIM-DM, 316–317*
    - *PIM-SM, 317–319*
    - *RP, 318, 319–321*
    - *SSM, 317*
- routing, 315–316
- service model, 310
- SSM, 309, 317
- traffic streams, 304–305
- UDP, 306
- unicasting versus, 304–305

multigroups, HSRP, 194–195

multihoming, BGP, 167–168

multilayer switches, 17–18

multiple frame transmission, redundant topologies, 56

multiplexing
- OFDM, 400
- spatial multiplexing, 404–405

multitenant mode, vManage, 290

MU-MIMO (Multi-User MIMO), 400, 401, 405–406

N

NAD (Network Access Devices)
- Central WebAuth, 489
- Wired NAD with Local WebAuth, 489

named ACL, 508, 510–511

namespaces, XML, 571–572

NAT (Network Address Translation), 205
- address types, 207–208
- benefits, 206
- configuring, 214
    - *dynamic NAT, 216*
    - *PAT, 215–217*
    - *static NAT, 215*
- disadvantages, 206–207
- dynamic NAT, 208–211, 214, 216
- implementation options, 208–209
- inside global addresses, 207
- inside local addresses, 207
- NAT overload. *See* PAT
- outside global addresses, 207
- outside local addresses, 207

- PAT, 209, 211–213, 215–217
- process, 205–206
- static NAT, 208–210, 214–215
- tuning, 217–218
- virtual interface, 213–214

native VLAN, 40, 41

NBAR (Network Based Application Recognition), 331–332

neighbors
- adjacencies
  - *EIGRP, 109–110*
  - *OSPF, 125–126*
- BGP
  - *neighbor relationships, 172*
  - *neighbor states, 171*
  - *neighbor tables, 169*
  - *OSPF neighbor states, 127–129*

Neighbors field, hello packets, 126

NETCONF (Network Configuration), 575–577
- configuration datastores, 580
- southbound API, 584
- SSH, 577–579

NetFlow, 378
- configuring, 382
- data analysis, 380–381
- event detectors, 616
- export data format, 381–382
- Flexible NetFlow, 384–387
- flow analyzers, 378
- flow collectors, 378
- flow creation in NetFlow cache, 379–380
- flow exporters, 378
- process, 378–379
- verifying, 382–384

Network Discovery API, 589

Network Plug and Play, DNA Center, 546–547

network services
- NAT, 205, 207–208
  - *benefits, 206*
  - *disadvantages, 206–207*
  - *dynamic NAT, 208–211, 214, 216*
  - *implementation options, 208–209*
  - *inside global addresses, 207*
  - *inside local addresses, 207*
  - *NAT overload. See PAT*
  - *outside global addresses, 207*
  - *outside local addresses, 207*
  - *PAT, 209, 211–213, 215–217*
  - *process, 205–206*
  - *static NAT, 208–210, 214–215*
  - *tuning, 217–218*
  - *virtual interface, 213–214*
- NTP, 205, 218
  - *access lists, 222–223*
  - *authentication, 222*
  - *clock offset, 219*
  - *configuring, 223–224*
  - *modes, 219–221*
  - *network delays, 219*
  - *security, 222–223*
  - *source addresses, 222*
  - *stratum example, 219*
  - *time packet exchanges, 219*
  - *verifying, 224–226*
  - *versions, 218–219*

networks
- assurance
  - *basic steps, 350*
  - *concepts, 349–350*
  - *connectivity testing, 353, 356–357*
  - *debug command, 357–360*
  - *diagnostics, 350–360*
  - *DNA Center Assurance, 541–542, 548–554*
  - *Flexible NetFlow, 384–387*
  - *IOS IP SLA, 360–365*
  - *logging services, 373–374*
  - *NetFlow, 378–387*
  - *ping command, 352–355*
  - *process, 351–352*
  - *SNMP, 376–378*
  - *SPAN, 365–372*
  - *syslog, 374–376*
  - *traceroute command, 355–357*
  - *traffic sniffers, 365*
- broadcast networks, 133, 134
- campus networks, architectures, 4–13
- configuration management tools (DevOps), 599–601
  - *Chef, 600–601*
  - *Puppet, 600*
  - *SaltStack, 601*
- converged networks
  - *bandwidth, 324*
  - *delay, 324–326*
  - *jitter, 325–327*
  - *network latency, 324–326*

  packet loss, 325, 327
  QoS, 324–327
Day N network automation, 547–548
delay, 324–325
  NTP, 219
  processing delay (variable), 325
  propagation delay (fixed), 326
  queuing delay (variable), 325
  serialization delay (fixed), 325
discovery process, DNA Center, 542–543
enterprise networks
  architectures, 4–13
  heirarchical LAN design, 1–4
hierarchies, 546
IBN, 537–539
LAN. *See also* VLAN
  VXLAN, 251, 262–267, 278
  WLAN, 407–417
latency, 324–326
Layer 2 networks
  Review Lab 1, 624–625
  Review Lab 2, 637
loopback networks, 134
managing
  DNA Center, 542–543
  Review Lab 1, 631–632
  SNMP, 565–567
NMS, 288
nonbroadcast networks, 133, 134
OSPF, 133
  broadcast networks, 133, 134
  loopback networks, 134
  nonbroadcast networks, 133, 134
  point-to-multipoint networks, 134
  point-to-point networks, 133, 134
overlay networks, 227, 273
  fabric technology, 274
  OMP, 298–301
  SD-WAN, 287
path isolation, 527
point-to-multipoint networks, 134
point-to-point networks, 133, 134
programmability, 555
  data models, 572–582
  device management, 565–567
  JSON, 567–570
  NETCONF, 575–580, 584
  Python, 555–565
  REST, 574–575
  RESTCONF, 580–582
  SNMP, 565–567
  XML, 570–572
  YANG, 573–574
provisioning, 546–547
SD-WAN, 285–294. *See also* WAN
  administration, 591
  analytics, 293–294
  API library/documentation, 592–598
  application performance optimization, 294–295
  application-aware routing, 294
  automation, 293–294
  certificate management, 591
  color, 297–298
  components, 287–294
  configuration, 591
  control plane, 288, 290–291
  data plane, 291–292
  device inventories, 591
  IaaS, 295
  management plane, 289–290
  monitoring, 591
  need for, 286–287
  OMP, 298–301
  orchestration plane, 288–289
  organization names, 296–297
  overlay networks, 287
  private IP addresses, 297
  public IP addresses, 297
  QoS, 294
  real-time monitoring, 591
  REST API, 290, 590–598
  routing, 299–301
  SaaS, 295
  service routes, 301
  site ID, 296
  solution example, 295–296
  system IP addresses, 296
  TLOC, 297, 299, 301
  troubleshooting tools, 591
  underlay networks, 287
  VPN, 298–299
security
  antispyware software, 472
  antivirus software, 472
  buffer overflow attacks, 467
  Cisco Talos website, 468
  content security, 470–471
  DDoS attacks, 466
  DMZ, 477
  DoS attacks, 466

- *endpoint security, 471–475*
- *exploits, 466*
- *firewalls, 472, 476–477*
- *hacking tools, 468*
- *IBNS, 484–490*
- *IPS, 468–469*
- *MACsec, 483–484*
- *malware, 467*
- *man-in-the-middle attacks, 467*
- *password attacks, 467*
- *personal firewalls, 472*
- *phishing attacks, 466–467*
- *reconnaissance attacks, 467*
- *reflection attacks, 466*
- *risk, 466*
- *social engineering attacks, 466*
- *spoofing attacks, 466*
- *threats, 466–468*
- *threatscape, 465–468*
- *TrustSec, 478–482*
- *vectors of data loss and exfiltration, 467*
- *VPN, 469–470*
- *vulnerabilities, 466*

topologies
- *fully meshed topologies, 233*
- *hub-and-spoke topologies, 233*

troubleshooting
- *basic steps, 350*
- *concepts, 349–350*
- *connectivity testing, 353, 356–357*
- *debug command, 357–360*
- *diagnostics, 350–360*
- *Flexible NetFlow, 384–387*
- *IOS IP SLA, 360–365*
- *logging services, 373–374*
- *NetFlow, 378–387*
- *ping command, 352–355*
- *process, 351–352*
- *SNMP, 376–378*
- *SPAN, 365–372*
- *syslog, 374–376*
- *traceroute command, 355–357*
- *traffic sniffers, 365*

underlay networks, 227, 272–273
- *fabric technology, 274*
- *SD-WAN, 287*

virtual networks, mapping users to, 278

VLAN, 40–41. *See also* LAN
- *802.1Q trunk ports, 38–43*
- *802.1X endpoint authentication, 487*
- *access ports, 35–36*
- *allowed VLAN, 41*
- *creating, 35*
- *DTP, 44–46*
- *end-to-end VLAN, 34*
- *inter-VLAN routing, 34, 50–53*
- *Layer 2 virtualization, 527–529*
- *local VLAN, 34–35*
- *native VLAN, 40, 41*
- *overview, 33–35*
- *routed switch ports, 53–54*
- *show interfaces switchport command, 38, 41–45*
- *show interfaces trunk command, 41–43*
- *show vlan brief command, 36–37*
- *show vlan command, 36–37*
- *show vlan id 2 command, 36–37*
- *show vlan name IT command, 36–37*
- *switchport mode dynamic auto command, 44*
- *switchport mode dynamic desirable command, 44*
- *switchport mode trunk command, 44*
- *switchport trunk encapsulation dot1q command, 41*
- *traffic patterns, 33–34*
- *VID, 40*

VPN, 469–470
- *GRE over IPsec transport mode, 242*
- *IPsec over GRE tunnel mode, 242–243*
- *IPsec VPN, 238–239*
- *SD-WAN, 298–299*
- *site-to-site VPN, 232–235, 242–249*

VXLAN, 251, 262–263
- *encapsulation, 263–265, 278*
- *decapsulation, 278*
- *gateways, 251*
- *VXLAN-GPO headers, 266–267*

WAN. *See also* SD-WAN
- *WAN Edge cloud routers, 287*
- *WAN Edge routers, 287–292, 297–298, 301*

wireless networks, mapping markings, 340

WLAN
- *autonomous AP deployments, 408–410*
- *centralized WLC deployments, 408, 410–415*
- *cloud-managed deployments, Meraki, 408, 416–417*

        FlexConnect deployments, 408, 415–416
        SD-Access deployments, 408
Next_Hop path attribute, BGP, 175
NFVIS (NFV Infrastructure Software), 524–527
NIC (Network Interface Cards), vNIC, 518–519
NMS (Network Management Systems), 288
no switchport command, 53–54
nodes, Chef, 611
noise, RF communications, 394
nonbroadcast networks, 133, 134
normal (nonbackbone) areas, OSPF, 133
normal mode (UDLD), 82
northbound API, 584–585
NOTIFICATION messages, BGP, 171
NSF (Nonstop Forwarding), 29–30
NTP (Network Time Protocol), 205, 218
    access lists, 222–223
    authentication, 222
    broadcast/multicast mode, 221
    client mode, 220
    clock offset, 219
    configuring, 223–224
    modes, 219–221
    network delays, 219
    peer mode, 221
    security, 222–223
    server mode, 220
    source addresses, 222
    stratum example, 219
    time packet exchanges, 219
    verifying
        associations, 225–226
        status, 224–225
    versions, 218–219
numbered ACL, 508–510
numbers data type, Python, 559, 561–562
NX-OS Python Shell, 558

O

objects
    SD-WAN, 591
    tracking, HSRP, 191–194
OEAP mode, AP, 427–428
OFDM (Orthogonal Frequency-Division Multiplexing), 400
off mode (VTP), 47
omnidirectional antennas, 429–430
OMP (Overlay Management Protocol), 298–301
on-roam scanning, wireless roaming, 435
OPEN messages, BGP, 169–170
OpenConfirm neighbor state, BGP, 171
OpenFlow, southbound API, 584
OpenSent neighbor state, BGP, 171
OpFlex, southbound API, 584
optional attributes, BGP
    nontransitive attributes, 176
    transitive attributes, 176
orchestration plane, SD-WAN, 288–289
organization names, SD-WAN, 296–297
origin authentication, IPsec, 237
origin path attribute, BGP, 175
OSPF (Open Shortest Path First), 121
    ABR, 133
        route filtering, 152
        route summarization, 148
    area hierarchies, 132
        backbone (area 0) areas, 133
        normal (nonbackbone) areas, 133
    AS, 122–123, 133
    ASBR, 133, 149
    bandwidth, adjusting, 143–144
    BDR election process, 134–136
    characteristics, 121–123
    costs, 143–144
    default routing, 146–147
    DR election process, 134–136
    filtering routes
        ABR, 152
        area ranges, 153
        distribute lists, 151–152, 154
        filter lists, 152–153
        summary addresses, 154–155
    functionality, 122
    LSA, 122–123
        propagation, 131
        route filtering, 153, 155
        SPF, 145
        types, 130
    LSDB, building, 127

- multiarea OSPF, 131–132
  - *configuring, 137–138*
  - *verifying functionality, 138–141*
- neighbors
  - *adjacencies, 125–126*
  - *states, 127–129*
- networks, 133
  - *broadcast networks, 133, 134*
  - *loopback networks, 134*
  - *nonbroadcast networks, 133, 134*
  - *point-to-multipoint networks, 134*
  - *point-to-point networks, 133, 134*
- packets, 129
  - *DBD packets, 129*
  - *encapsulation, 129*
  - *hello packets, 125–126, 129*
  - *LSAck packets, 129*
  - *LSR packets, 129*
  - *LSU packets, 129*
- passive interfaces, 145
- process ID, 123–124
- router ID, 124–125
- routing, 131–132
  - *backbone routers, 133*
  - *BDR election process, 134–136*
  - *default routing, 146–147*
  - *DR election process, 134–136*
  - *filtering options, 152*
  - *internal routers, 133*
  - *route filtering tools, 151–155*
  - *summarizing routes, 147–150*
- single-area OSPF, 131–132
- SPF, 144–145
- timers, 136–137
- topological (link-state) databases, 122
- verifying functionality, 138–141

OSPFv3, 155–156
- ABR, 156
- ASBR, 156
- configuring, 157–160
- LSA, 156–157
- verifying, 161–162

outbound ACL, 511

outside global addresses, NAT, 207

outside local addresses, NAT, 207

overlay networks, 227, 273
- fabric technology, 274
- OMP, 298–301
- SD-WAN, 287

P

packets
- acknowledge packets, 109
- DBD packets, OSPF, 129
- EIGRP packets, 108–109
- encapsulation, OSPF, 129
- forwarding, 19
  - *control plane, 19*
  - *data plane, 19*
  - *Layer 2 forwarding tables, 18*
  - *Layer 3 forwarding tables, 18*
  - *NSF, 29–30*
- hello packets, 108, 125–126, 129
- loss, 325, 327
- LSAck packets, OSPF, 129
- LSR packets, OSPF, 129
- LSU packets, OSPF, 129
- OSPF
  - *encapsulation, 129*
  - *types, 129*
- query packets, 109
- RADIUS packet flow, 497–498
- reply packets, 109
- request packets, 109
- switching, 20
  - *CEF, 21–22, 24–27*
  - *centralized switching, 26–27*
  - *Cisco Supervisor Engine, 28–29*
  - *dCEF, 26–27*
  - *fast switching, 20–23*
  - *hardware redundancy, 28–29*
  - *Layer 2 switches, 15–17*
  - *Layer 3 switches, 17–19*
  - *process switching, 20, 22–23*
  - *RPR, 28*
  - *RPR+, 28–29*
  - *SDM templates, 30–31*
  - *SSO, 29*
- TACACS+ packet flow, 498
- update packets, 108

PAgP (Port Aggregation Protocol), 93–95
- Auto mode, 95
- configuring, 98–99
- Desirable mode, 95
- Silent mode, 95
- static-on mode, 95

partial mesh topologies, 233

passive interfaces, OSPF, 145

passive mode (LACP), 94

- **passive scanning, wireless roaming, 435**
- **passwords**
  - configuring, 494–495
  - EIGRP, 110
  - enable secret password, 492–493
  - line passwords, 493–494
  - password attacks, 467
  - PAT (Port Address Translation), 209, 211–213, 215–217
- **PATCH method, RESTCONF, 581**
- **path attributes, BGP, 175**
  - discretionary attributes, 175
  - mandatory attributes, 175
- **path isolation, networks, 527**
- **path manipulation, STP, 74–76**
- **path selection**
  - BGP, 173–174
  - EIGRP, 115–117
- **Path Trace API, 589**
- **path vectors, BGP, 165–166**
- **PCP (Priority Code Points), 40**
- **PEPS (Python Style Guide), 559**
- **performance, applications, 294–295**
- **personal firewalls, 472**
- **PETR (Proxy ETR), 255**
- **phishing attacks, 466–467**
- **physical servers, 518**
- **pillars, SaltStack, 614**
- **PIM (Protocol Independent Multicasts), 316**
  - BIDIR-PIM mode, 317
  - PIM-DM, 316–317
  - PIM-SM, 317–319
  - RP, 318, 319
    - *auto-RP, 320–321*
    - *BSR, 320–321*
    - *candidate RP, 320*
    - *RP-mapping agents, 320*
    - *static RP, 319*
  - SSM, 317
- **ping command, 352**
  - connectivity testing, 353
  - extended ping command, 353–355
  - options, 353
- **PITR (Proxy ITR), 255**
- **playbooks, Ansible, 602, 604**
  - components, 604–605
  - example, 604–605
  - executing, 606
  - inventory files, 605–606
  - pound sign (#), 605
  - terms, 604
- **plays, Ansible, 604**
- **PnP API, 588**
- **point-to-multipoint networks, 134**
- **point-to-point networks, 133, 134**
- **point-to-point VPN connections, individual, 232–233**
- **polarization, CEF, 21**
- **policies**
  - EEM policies, 615–620
  - maps, 278, 514
- **policing tools, QoS, 330, 340–341**
- **port channel min-links, LACP, 103**
- **ports**
  - 802.1Q trunk ports, 38–39
    - *configuring, 41*
    - *tagging process, 39–40*
    - *verifying, 41–43*
  - access ports, VLAN, 35–36
  - aggregating, EtherChannel, 91
    - *configuring, 95–96*
    - *LACP, 93–94, 97–98, 101–104*
    - *load balancing, 96–97*
    - *need for, 91–93*
    - *PAgP, 93–95, 98–99*
    - *verifying, 99–101*
  - alternate ports, 64
  - AUX ports, syslog messages, 374
  - backup ports, 64
  - blocking state, 63
  - BPDU frames, 65
  - designated ports, 64
  - disabled ports, 64
  - disabled state, 63, 64
  - discarding state, 65
  - DTP, 44–45
    - *combinations, 45*
    - *configuring, 45–46*
    - *verifying status, 44–46*
  - edge ports, 66
  - forwarding state, 63–65
  - hot-standby ports, LACP, 101–102
  - learning state, 63, 65
  - listening state, 63
  - MST port priorities, 89
  - no switchport command, 53–54
  - prioritizing, LACP, 103

- root ports, 64
  - *designated port elections, 62–63*
  - *elections, 60–63*
- routed switch ports, 53–54
- RSTP
  - *port roles, 64*
  - *port states, 65*
- show interfaces switchport command, 38, 41–45
- show interfaces trunk command, 41–43
- SPAN
  - *ERSPAN, 370–372*
  - *local SPAN, 366–368*
  - *overview, 365–366*
  - *remote SPAN, 368–370*
- STP
  - *port costs, 61–62, 75–76*
  - *port roles, 71*
  - *port states, 56, 63–64, 71*
- STP PortFast, 77–79
- switchport mode access command, 45
- switchport mode dynamic auto command, 44
- switchport mode dynamic desirable command, 44
- switchport mode trunk command, 44–45
- VTY ports, syslog messages, 374

POST action, REST API, 585, 586

POST method, RESTCONF, 581

Postman, vManage and REST API operations, 596–598

pound sign (#), Ansible playbooks, 605

PQ (Priority Queuing), 343

preempting HSRP, 189–190

presence analytics, 444

printing strings, Python, 560

prioritizing
- AP failovers, 424
- bridges, STP, 71–73
- HSRP, 189
- port priority, LACP, 103
- system priority, LACP, 103

private IP addresses, SD-WAN, 297

probes, wireless roaming
- broadcast probes, 435
- directed probes, 435

process ID, OSPF, 123–124

process switching, 20, 22–23

processing delay (variable), 325

producers, 347

programmability, networks, 555
- data encoding, 567
  - *JSON, 567–570*
  - *XML, 570–572, 579*
- data models, 572–582
- NETCONF, 575–577
  - *configuration datastores, 580*
  - *southbound API, 584*
  - *SSH, 577–579*
- Python
  - *concepts, 555–556*
  - *data types, overview, 559*
  - *describe conditionals, 562–563*
  - *executing code, 556–557*
  - *helper utilities, 557–558*
  - *idiomatic Python (being Pythonic), 558–559*
  - *PEPS, 559*
  - *Python Shell, 556–558*
  - *writing scrypts, 557, 564–565*
- REST, 574–575
- RESTCONF, 580
  - *GET example, 581–582*
  - *methods, 580–581*
- SNMP, 565–567
- YANG, 573–574

propagation delay (fixed), 326

proprietrary load-balancing, wireless roaming, 434–435

provisioning networks, 546–547

proxy minions, SaltStack, 613

proxy-agent configuration, 599

PSK authentication, 446–449

public IP addresses, SD-WAN, 297

Puppet
- architectures, 606
- catalogs, 607
- classes, 607
- concepts, 607–608
- configuration management, 600
- configuration script, 608–610
- example, 608–610
- manifests, 607
- modules, 607
- resources, 607
- system management, 607–610

push model, Ansible, 603–604

PUT action, REST API, 585, 586

PUT method, RESTCONF, 581

PVST+ (Per VLAN Spanning Tree+), 55

Python

- Boolean data type, 559, 562
- concepts, 555–556
- data types, overview, 559
- describe conditionals, 562–563
- dictionary data type, 559
- executing code, 556–557
- file data type, 559
- helper utilities, 557–558
- idiomatic Python (being Pythonic), 558–559
- list data type, 559
- numbers data type, 559, 561–562
- PEPS, 559
- Python Shell, 556–558
- scripts, writing, 557, 564
  - *entry points, 564–565*
  - *shebangs, 564*
- string data type, 559–560
  - *built-in methods, 561*
  - *concatenating strings, 560*
  - *printing strings, 560*
- vManage, REST API operations, 594–596

Q

QoS (Quality of Service), 323

- AutoQoS, 347
- best effort QoS model, 327, 328
- classification tools, 329–332, 334
  - *802.1p Class of Service, 334–336*
  - *802.11 Wireless QoS: 802.11e, 336–337*
- congestion management, 330, 342
  - *congestion avoidance, 344–345*
  - *scheduling tools, 342*
- converged networks, 324–327
- DiffServ model, 328–329
- IntServ model, 328
- Layer 2 switches, 17
- Layer 3 switches, 18
- link-specific tools, 330
- LLQ, 342–344
- marking tools, 329–330, 332–334
  - *802.11 Wireless QoS: 802.11e, 336–337*
  - *802.1p Class of Service, 334–336*
  - *Layer 2, 334–337*
  - *Layer 3, 337–340*
- MQC, 346–347
- NBAR, 331–332
- need for, 323–324
- policies, 345
  - *AutoQoS, 347*
  - *defining, 345–346*
  - *DNA Center application policies, 347–348*
  - *MQC, 346–347*
- policing tools, 330, 340–341
- re-marking tools, 330, 340–341
- scheduling tools, 330, 342
- SD-WAN, 294
- shaping tools, 330, 340–341
- trust boundaries, QoS, 333–334
- Type of Service, 337–338

querier election mechanism, IGMPv2, 312

queries, IGMPv2

- group-specific queries, 312
- query-interval response time, 312

query packets, 109

queuing

- CBWFQ, 343–344
- CQ, 343
- delay (variable), 325
- FIFO, 343
- LLQ, 342–344
- PQ, 343
- round-robin queuing, 342
- strict-priority queuing (traffic starvation), 342
- tail drops, 344–345
- WFQ, 342–344

R

RADIUS, 496–497

- configuring, 500–501
- packet flow, 497–498

rapid convergence, EIGRP, 106

Rapid PVST+, 55

RBAC (Role-Based Access Control), 289

RCPI (Received Channel Power Indicators), 394

RD (Reported Distance), EIGRP, 116

reactor (respond), SaltStack, 613

real-time monitoring, SD-WAN, 591

recipes, Chef, 611

reconnaissance attacks, 467

redundancy

- AAA framework, 496
- CRC, 20
- default gateway redundancy, 183–184
- FHRP
  - *default gateway failovers, 185*
  - *failover process, 185–186*
  - *GLBP, 185*
  - *HSRP, 185, 186–199*
  - *operations, 184–185*
  - *VRRP, 185*
- first-hop redundancy, Review Lab 1, 627–630
- hardware redundancy
  - *Cisco Supervisor Engine, 28–29*
  - *RPR, 28*
  - *SSO, 29*
- HSRP, 186
  - *active routes, 187*
  - *forwarding, 188*
  - *groups, 187–189*
  - *load balancing, 194*
  - *multigroups, 194–195*
  - *object tracking, 191–194*
  - *preempting, 189–190*
  - *prioritizing, 189*
  - *standby groups, 186–187*
  - *standby routes, 187*
  - *states, 191*
  - *timers, 190–191*
- Layer 2 redundancy, 55
- MHSRP, load balancing, 194
- RPR, 28
- RPR+, 28–29
- topologies
  - *broadcast storms, 56*
  - *MAC database instability, 56*
  - *multiple frame transmission, 56*
- VRRP, 185, 199–200
  - *authentication, 201*
  - *configuring, 201–203*

reflection attacks, 466

regions, MST, 84–85

registering, endpoints, 278

Reliability (K4, K5) metric, EIGRP, 110

re-marking tools, QoS, 330, 340–341

remote SPAN, 368–370

reply packets, 109

reporting, Baseline Deviation Reporting, 551

request packets, 109

resource data, SD-WAN, 591

resources

- Chef, 611
- Puppet, 607

respond (reactor), SaltStack, 613

response codes, REST API, 585–586

REST (Representational State Transfer), 574–575

REST API, 584–585

- authentication, 589
- communication process, 585
- CRUD operations, 585–586
- DNA Center, 587–590
- HTTP status codes, 586
- integration API, 590
- Know Your Network REST request paths, 588
- multivendor support, 589–590
- operational tools, 589
- response codes, 585–586
- SD-WAN, 290
  - *arrays, 591*
  - *objects, 591*
  - *resource collections, 591*
  - *resource data, 591*
  - *scalar data, 591*
- security, 586–587
- site management API, 588
- vManage, 590–598
  - *Postman, REST API operations, 596–598*
  - *Python, REST API operations, 594–596*

RESTCONF, 580

- GET example, 581–582
- methods, 580–581

retrospective alerting (cloud recall), 475

retry counts, wireless roaming, 434

Review Lab 1

- addressing tables, 623
- building networks, 624
- configuring
  - *basic settings, 624*
  - *routing protocols, 625–627*
  - *secure access, 630–631*
- first-hop redundancy, 627–630
- host support, 624–625

- interface addressing, 624
- IP SLA functionality, 627–630
- Layer 2 networks, configuring, 624–625
- network management, 631–632
- objective, 621
- tasks, 623
- topologies, 621

Review Lab 2
- addressing tables, 634–635
- building networks, 635
- configuring
  - *basic settings, 635*
  - *secure access, 637–638*
  - *static routing, 636–637*
  - *VRF, 636–637*
- Layer 2 networks, configuring, 637
- objective, 633
- tasks, 635
- topologies, 633

RF (Radio Frequency) communications, 389
- amplitude, 391–392
- antennas, 397–399
- decibels (dB), 395–397
- EIRP, 398–399
- free path loss, 392–393
- frequencies, 390
- noise, 394
- RCPI, 394
- RSSI, 393–394, 434
- SNR, 394–395
- spectrums, 389–390
- watts (W), 395, 396
- wavelengths, 391

RIB (Routing Information Bases)
- CEF, 24
- NSF, 29

risk, network security, 466

RLOC addresses, 252–253

roaming, wireless, 434
- active scanning, 435
- auto-anchor mobility (guest tunneling), 441–442
- background scanning, 435
- broadcast probes, 435
- directed probes, 435
- Layer 2 roaming, 437–439
- Layer 3 roaming, 439–441
- maximum data retry counts, 434
- mobility groups, 435–436
- passive scanning, 435
- proprietary load-balancing, 434–435
- on-roam scanning, 435
- RSSI, 434
- SNR, 434
- types, 436–441

rogue detector mode, AP, 428

roles, Ansible, 604

root bridges
- bridge priorities, 59–60
- elections, 59–60
- Extended System ID, 60
- MAC addresses, 60

Root Guard, 79–80

root ports, 64
- designated port elections, 62–63
- elections, 60–63

round-robin queuing, 342

routed switch ports, 53–54

router ID
- hello packets, 126
- OSPF, 124–125

routing
- ABR
  - *OSPF, 133, 148, 152*
  - *OSPFv3, 156*
- ALT routers, 255
- application-aware routing, SD-WAN, 294
- ASBR
  - *OSPF, 133, 149*
  - *OSPFv3, 156*
- backbone routers, OSPF, 133
- BDR
  - *hello packets, 126*
  - *OSPF BDR election process, 134–136*
- BGP, 166–167
  - *interdomain routing, 163–164*
  - *route aggregation, 175*
  - *route selection, 174*
- BSR, PIM, 320–321
- Cisco ISR 4000 routers, 526
- DR
  - *hello packets, 126*
  - *OSPF DR election process, 134–136*
- EIGRP, 105–106
  - *AS numbers, 109*
  - *authentication, 110*
  - *ECMP, 106*
  - *encapsulation, 106–107*

- *FD, 116*
- *feasible successors, 107*
- *K values, 109–115*
- *load balancing, 106, 117–119*
- *loop-free, classless routing protocol, 106*
- *loop-free path selection, 116–117*
- *metrics (K values), 109–115*
- *multi-address family support, 106*
- *neighbor adjacencies, 109–110*
- *operation overview, 107–108*
- *packets, 108–109*
- *passwords, 110*
- *path selection, 115–117*
- *rapid convergence, 106*
- *RD, 116*
- *reduced bandwidth use, 106*
- *reliable transport protocol, 106–107*
- *subnets, 109*
- *successor routes, 107*

ETR, 254

external routers, inter-VLAN routing, 51–52

filtering tools
- *ABR, 152*
- *area ranges, 153*
- *distribute lists, 151–152, 154*
- *filter lists, 152–153*
- *OSPF, 151–155*
- *summary addresses, 154–155*

fusion routers, 282–283

GRE, 227–228
- *configuring, 228–230*
- *GRE over IPsec, crypto maps, 243–245*
- *GRE over IPsec Site-to-Site VPN, 243–244*
- *GRE over IPsec transport mode, 242*
- *GRE over IPsec, Tunnel IPsec Profiles, 245–246*
- *IPsec over GRE tunnel mode, 242–243*

HSRP, 185, 186
- *active routes, 187*
- *authentication, 195–196*
- *configuring, 197–199*
- *forwarding, 188*
- *groups, 187–189*
- *load balancing, 194*
- *multigroups, 194–195*
- *object tracking, 191–194*
- *preempting, 189–190*
- *prioritizing, 189*
- *standby groups, 186–187*
- *standby routes, 187*
- *states, 191*
- *timers, 190–191*
- *versions, 196–197*

interdomain routing, BGP, 163–164

internal routers, OSPF, 133

inter-VLAN routing, 34, 50–51
- *encapsulation dot1q command, 51–52*
- *external routers, 51–52*
- *subinterfaces, 51–52*
- *SVI, 52–53*

IP routing
- *LISP versus, 252*
- *BGP tables, 169*

ITR, 253–254

Layer 3 access layer (router-based) design, 7–8, 11

LISP, 251, 252
- *ALT routers, 255*
- *components, 253*
- *control plane, 258, 275*
- *data plane, 255–258, 275*
- *EID addresses, 252–253, 255*
- *ETR, 254*
- *forwarding, 278*
- *host mobility, 259–262*
- *IP routing versus, 252*
- *ITR, 253–254*
- *mapping LISP instances to VRF instances, 278*
- *MR, 254*
- *MS, 254*
- *PETR, 255*
- *PITR, 255*
- *RLOC addresses, 252–253*

loop-free, classless routing protocol, 106

MHSRP

multicasting, 315–316

OMP routes, SD-WAN, 299–301

OSPF, 131–132
- *ABR, 152*
- *area ranges, 153*
- *backbone routers, 133*
- *BDR election process, 134–136*
- *default routing, 146–147*
- *distribute lists, 151–152, 154*
- *DR election process, 134–136*
- *filter lists, 152–153*
- *filtering options, 152*
- *internal routers, 133*
- *route filtering tools, 151–155*

    route summarization, 147–150
    SPF, 144–145
    summary addresses, 154–155
  PETR, 255
  PITR, 255
  router priority, hello packets, 126
  SD-WAN, 299–301
  service routes, SD-WAN, 301
  SPF, OSPF, 144–145
  static routing, Review Lab 2, 636–637
  subinterfaces, configuring, 51–52
  summarizing routes
    BGP route aggregation, 175
    OSPF, 147–150
  TLOC routes, SD-WAN, 301
  VRRP, 185, 199–200
    authentication, 201
    configuring, 201–203
  WAN Edge cloud routers, 287
  WAN Edge routers, 287–292, 297–298, 301

routing event detectors, 617

routing protocols, Review Lab 1, configuring, 625–627

RP (Rendezvous Points), 318, 319
  auto-RP, 320–321
  BSR, 320–321
  candidate RP, 320
  RP-mapping agents, 320
  static RP, 319

RPF(Reverse-Path Forwarding), 315–316

RPR (Route Processor Redundancy), 28

RPR+, 28–29

RSA
  encrypted nonces, 239
  IPsec, 237
  signatures, 238

RSSI (Received Signal Strength Indicators), 393–394, 434

RSTP (Rapid Spanning Tree Protocol), 55, 64
  configuring, 68–69, 76–77
  convergence, 67
  edge ports, 66
  link type, rapid transition to forwarding state, 66
  port roles, 64
  port states, 65
  rapid transition to forwarding state, 66
  show-spanning tree command, 69–71
  synchronizing, 66–67
  topology changes, 67–68
  verifying, bridge ID, 69–70

RSVP (Resource Reservation Protocol), 328

S

SA (Security Associations), IPsec, 239–240

SaaS (Software as a Service), SD-WAN, 295

SaltStack, 612
  architectures, 612–614
  configuration management, 601
  configuration script, 614–615
  examples, 614–615
  grains, 613
  masters, 612
  mines, 614
  minions, 612
  monitor (beacons), 613
  pillars, 614
  proxy minions, 613
  respond (reactor), 613
  SSH, 612
  states (formulas), 613
  system management, 612–615

scalability, AAA framework, 496

scalar data, SD-WAN, 591

scanning, wireless roaming
  active scanning, 435
  background scanning, 435
  passive scanning, 435
  on-roam scanning, 435

scheduling tools, QoS, 330, 342

schema trees, 573

scripts
  Chef configuration script, 610
  Puppet configuration script, 608–610
  Python scripts, writing, 557, 564
    entry points, 564–565
    shebangs, 564
  SaltStack configuration script, 614–615
  tcl scripts, EEM, 616, 618–620

SD-Access, 11–12, 269–270
  benefits, 270

- Cisco DNA Center, 270–272
- design, 11–12
- fabric technology, 269, 272–274
  - *components, 277–279*
  - *control plane, LISP, 275*
  - *data plane, LISP, 275*
  - *policy plane, 275–276*
- fusion routers, 282–283
- overview, 272–274
- shared services, 281–283
- topologies, 282–283
- WLAN deployments, 408

SDM templates, 30–31

SDN controllers, API, 584

SDP/SAP blocks (224.2.0.0/16), 309

SD-WAN (Software-Defined WAN), 285–286
- administration, 591
- analytics, 293–294
- API library/documentation, 592–598
- applications
  - *application-aware routing, 294*
  - *performance optimization, 294–295*
- architectures, 287–294
- automation, 293–294
- certificate management, 591
- color, 297–298
- components, 287–294
- configuration, 591
- control plane, 288, 290–291
- data plane, 291–292
- device inventories, 591
- IaaS, 295
- management plane, 289–290
- monitoring, 591
- need for, 286–287
- OMP, 298–301
- orchestration plane, 288–289
- organization names, 296–297
- overlay networks, 287
- private IP addresses, 297
- public IP addresses, 297
- QoS, 294
- real-time monitoring, 591
- REST API, 290
  - *arrays, 591*
  - *objects, 591*
  - *resource collections, 591*
  - *resource data, 591*
  - *scalar data, 591*
- routing, 299–301
- SaaS, 295
- service routes, 301
- site ID, 296
- solution example, 295–296
- system IP addresses, 296
- TLOC, 297, 299, 301
- troubleshooting tools, 591
- underlay networks, 287
- VPN, 298–299

SE-Connect mode, AP, 428

secure access, configuring
- Review Lab 1, 630–631
- Review Lab 2, 637–638

security
- AAA framework, 497
  - *accounting, 495, 503*
  - *authentication, 495, 496, 498–499*
  - *authorization, 495, 501–503*
  - *configuring local users, 499–500*
  - *enabling, 499–500*
  - *flexibility, 496*
  - *granular access control, 496*
  - *overview, 495–496*
  - *RADIUS, 496–498, 500–501*
  - *redundancy, 496*
  - *scalability, 496*
  - *TACACS+, 497, 498, 501*
- access control, 491
- ACL
  - *extended ACL, 508, 513*
  - *inbound ACL, 511–512*
  - *interfaces, applying to, 511–513*
  - *named ACL, 508, 510–511*
  - *numbered ACL, 508–510*
  - *outbound ACL, 511*
  - *overview, 504–505*
  - *standard ACL, 507–508, 512*
  - *wildcard masking, 505–507*
- CoPP, 513–514
  - *class maps, 514*
  - *configuring, 514–515*
  - *policy maps, 514*
  - *service policies, 514*
- DNA Center, 536–537
- encryption, symmetric encryption algorithms, 237–238
- ESP, 236
- infrastructure security
  - *AAA framework, 495–503*
  - *ACL, 504–513*
  - *CoPP, 513–515*

    - *device access security, 492–495*
    - *enable secret password, 492–493*
    - *line passwords, 493–494*
    - *passwords, 494–495*
    - *secure access control, 491*
    - *usernames, 494–495*
  - IPS, 478
  - IPsec, 227
    - *3DES algorithm, 237*
    - *AES, 237–238*
    - *AH, 236*
    - *antireplay protection, 237*
    - *confidentiality, 236*
    - *data integrity, 237*
    - *DES algorithm, 237*
    - *ECDSA signatures, 239*
    - *ESP, 236*
    - *GRE over IPsec, crypto maps, 243–245*
    - *GRE over IPsec Site-to-Site VPN, 243–244*
    - *GRE over IPsec transport mode, 242*
    - *GRE over IPsec, Tunnel IPsec Profiles, 245–246*
    - *HMAC, 238*
    - *IKE, 236, 239–242*
    - *IPsec over GRE tunnel mode, 242–243*
    - *key management, 237*
    - *origin authentication, 237*
    - *overview, 235–236*
    - *PSK, 238*
    - *RSA, 237*
    - *RSA encrypted nonces, 239*
    - *RSA signatures, 238*
    - *SA, 239–240*
    - *services, 236–239*
    - *site-to-site VPN, 232–235, 242–249*
    - *site-to-site VTI over IPsec, 247–249*
    - *VPN, 238–239*
    - *VTI, 243*
  - ISAKMP
    - *crypto maps, 244–245*
    - *GRE over IPsec, Tunnel IPsec Profiles, 245–246*
    - *IKE, 239–241*
    - *site-to-site VTI over IPsec, 247, 249*
  - networks
    - *antispyware software, 472*
    - *antivirus software, 472*
    - *buffer overflow attacks, 467*
    - *Cisco Talos website, 468*
    - *content security, 470–471*
    - *DDoS attacks, 466*
    - *DMZ, 477*
    - *DoS attacks, 466*
    - *endpoint security, 471–475*
    - *exploits, 466*
    - *firewalls, 472, 476–477*
    - *hacking tools, 468*
    - *IBNS, 484–490*
    - *IPS, 468–469*
    - *MACsec, 483–484*
    - *malware, 467*
    - *man-in-the-middle attacks, 467*
    - *password attacks, 467*
    - *personal firewalls, 472*
    - *phishing attacks, 466–467*
    - *reconnaissance attacks, 467*
    - *reflection attacks, 466*
    - *risk, 466*
    - *social engineering attacks, 466*
    - *spoofing attacks, 466*
    - *threats, 466–468*
    - *threatscape, 465–468*
    - *TrustSec, 478–482*
    - *vectors of data loss and exfiltration, 467*
    - *VPN, 469–470*
    - *vulnerabilities, 466*
  - NTP, 222–223
  - passwords
    - *configuring, 494–495*
    - *enable secret password, 492–493*
    - *line passwords, 493–494*
  - RADIUS, 496–497, 500–501
  - REST API, 586–587
  - SA, IPsec, 239–240
  - SGT, 487
  - symmetric encryption algorithms, 237–238
  - TACACS+, 497
    - *configuring, 501*
    - *packet flow, 498*

- **serialization delay (fixed), 325**
- **server mode (VTP), 47**
- **servers**
  - Chef servers, 610
  - Cisco E-Series servers, 526
  - Cisco UCS C-Series servers, 526
  - EEM servers, 615–616
  - physical servers, 518
  - virtualization, 517–520
- **service policies, CoPP, 514**
- **service routes, SD-WAN, 301**

SGT (Security Group Access), 802.1X endpoint authentication, 487

shaping tools, QoS, 330, 340–341

shared services, SD-Access, 281–283

shared trees, 314–315

shebangs, Python scripts, 564

show interfaces switchport command, 41

Silent mode (PAgP), 95

simple entries, dynamic NAT, 211

simplified campus design, 9–10
- Layer 2 looped design, 10
- Layer 2 loop-free design, 10
- StackWise, 9–10
- VSS, 9–11

single-area OSPF, 131–132

single-homed connections, BGP, 168

site ID, SD-WAN, 296

site management API, 588

site scope, 347

site-to-site VPN, 232
- configuring, 242–249
- DMVPN, 234
- FlexVPN, 235
- fully meshed network topologies, 233
- GRE over IPsec, site-to-site VPN, 243–244
- hub-and-spoke network topologies, 233
- individual point-to-point VPN connections, 232–233
- joined hub-and-spoke topologies, 233
- partial mesh topologies, 233
- tiered hub-and-spoke topologies, 233
- topologies, 232–233
- VTI over IPsec, 247–249

SLA (Service-Level Agreements), IOS IP SLA, 360–365

sniffer mode, AP, 428

sniffers, traffic, 365

SNMP (Simple Network Management Protocol), 376–378, 565–567, 616

SNR (Signal-to-Noise Ratio), 394–395, 434

social engineering attacks, 466

software
- antispyware software, 472
- antivirus software, 472
- image management, 544–545
- SaaS, SD-WAN, 295

software-defined access (SD-access) design, 11–12, 269–270
- Cisco DNA Center, 270–272
- fabric technology, 269, 272–274
  - *components, 277–279*
  - *control plane, LISP, 275*
  - *data plane, LISP, 275*
  - *policy plane, 275–276*
- fusion routers, 282–283
- NFVIS, 524–527
- overview, 272–274
- shared services, 281–283
- topologies, 282–283

source addresses, NTP, 222

source trees, 313, 315

Source-Specific Multicast blocks (232.0.0.0/8), 309

southbound API, 584

spaces (indentation), idiomatic Python (being Pythonic), 559

SPAN (Switched Port Analyzers)
- ERSPAN, 370–372
- local SPAN, 366–368
- overview, 365–366
- remote SPAN, 368–370

spatial multiplexing, 404–405

Speak state, HSRP, 191

spectrums, RF communications, 389–390

SPF (Shortest Path First), OSPF, 144–145

spine and leaf architectures, 12–13

split MAC, centralized WLC deployments, 411–413

split tunneling, 416

spoofing attacks, 466

spyware, antispyware software, 472

SSH (Secure Shell)
- NETCONF over SSH, 577–579
- SaltStack, 612

SSM (Source-Specific Multicasting), 309, 317

SSO (Stateful Switchovers), 29

stability, STP stability mechanisms, 77–83

StackWise, simplified campus design, 9–10

standalone mode, AP, 427

standard ACL, 507–508, 512

standby groups, HSRP, 186

Standby state, HSRP, 191

starvation, traffic, 342

states (formulas), SaltStack, 613

static NAT, 208–210, 214–215

static routing, Review Lab 2, 636–637

static RP, 319

static-on mode (PAgP), 95

status codes, HTTP, 586

STP (Spanning Tree Protocol), 55
- BID, 59, 60
- BPDU, 57–59
- BPDU Guard, 77–79
- bridge ID, verifying, 69–70
- bridges, prioritizing, 71–73
- bridging loops, 55–56
- EtherChannel, 91–93
- Layer 2 access layer (STP-based) design, 6–7
- MST, 83–84
  - *configuring, 87–89*
  - *instances, 85–86*
  - *path costs, 89*
  - *port priorities, 89*
  - *regions, 84–85*
  - *topologies, 89*
  - *verifying, 88*
- operations, 57
- overview, 55–56
- path manipulation, 74–76
- ports
  - *port costs, 61–62, 75–76*
  - *port roles, 71*
  - *port states, 56, 63–64, 71*
- root ports, 60–63
- root bridges, elections, 59–60
- Root Guard, 79–80
- root ports
  - *designated port elections, 62–63*
  - *elections, 60–63*
- RSTP, 55, 64
  - *configuring, 68–69, 76–77*
  - *convergence, 67*
  - *edge ports, 66*
  - *link type, rapid transition to forwarding state, 66*
  - *port roles, 64*
  - *port states, 65*
  - *rapid transition to forwarding state, 66*
  - *show-spanning tree command, 69–71*
  - *synchronizing, 66–67*
  - *topology changes, 67–68*
  - *verifying, 69–70*
- show-spanning tree command, 69–71
- stability mechanisms, 77–83
- STP Loop Guard, 80–81
- STP PortFast, 77–79
- UDLD, 82–83
- verifying
  - *bridge ID, 69–70*
  - *RSTP, 69–70*
  - *STP mode, 76–77*

strict-priority queuing (traffic starvation), 342

string data type, Python, 559–560
- built-in methods, 561
- concatenating strings, 560
- printing strings, 560

stub area flags, hello packets, 126

subinterfaces, inter-VLAN routing, 51–52

subnets
- EID, border node, fabric devices, 278
- EIGRP, 109
- host mobility
  - *across subnets, 260–261*
  - *example of, 260–262*
  - *extended subnets, 259–260*

subnetwork broadcasts, 422

subordinate AP, Cisco Mobility Express, 422

successor routes, EIGRP, 107

summarizing routes
- BGP route aggregation, 175
- OSPF, 147–148
  - *ABR, 148*
  - *ASBR, 149*

summary addresses, OSPF route filtering, 154–155

SVI (Switched Virtual Interfaces)
- configuring, 53
- inter-VLAN routing, 52–53

SWIM API, 588

switches
  Catalyst 9800 Series embedded wireless controllers, 419–420
  multilayer switches, 17–18
  no switchport command, 53–54
  routed switch ports, 53–54
  show interfaces switchport command, 38, 41–45
  show interfaces trunk command, 41–43
  switchport mode access command, 45
  switchport mode dynamic auto command, 44
  switchport mode dynamic desirable command, 44
  switchport mode trunk command, 44–45
switching
  central, 416
  local, 416
switching, packet, 20
  CEF, 22, 24–26
    centralized switching, 26–27
    dCEF, 26–27
    polarization, 21
  centralized switching, 26–27
  Cisco Supervisor Engine, 28–29
  dCEF, 26–27
  fast switching, 20–23
  hardware redundancy, 28–29
  Layer 2 switches
    ACL, 17
    CAM tables, 15–17
    Layer 2 forwarding tables, 16
    MAC address tables, 17
    QoS, 17
    TCAM tables, 17
    traffic switching process, 16–17
    unknown unicast flooding, 15
  Layer 3 switches
    ACL, 18
    CAM tables, 18, 19
    QoS, 18
    TCAM tables, 18
    traffic switching process, 17–19
  process switching, 20, 22–23
  RPR, 28
  SDM templates, 30–31
  SSO, 29
switchport trunk allowed vlan command, 41
switchport trunk encapsulation dot1q command, 41
switchport trunk native vlan command, 41
symmetric encryption algorithms, 237–238
synchronizing, RSTP, 66–67
syslog, 374–375
  event detectors, EEM, 616
  message format, 375
  servers, 374
  severity levels, 375–376
system IP addresses, SD-WAN, 296
system management
  Ansible
    Ansible Tower, 602
    control station, 602
    inventory files, 602
    modules, 602, 604
    operations, 603–604
    playbooks, 602, 604–606
    plays, 604
    push model, 603–604
    roles, 604
    system management tools, 603
    tasks, 604
    YAML files, 602
  Chef, 607–610
    clients, 611
    concepts, 610–611
    configuration script, 610
    cookbooks, 611
    examples, 611–612
    nodes, 611
    recipes, 611
    resources, 611
    servers, 610
  EEM, 615
    applets, 616–619
    architectures, 615
    Cisco IOS CLI, writing EEM policies, 618–619
    event detectors, 615–617
    policies, 615–620
    servers, 615–616
    tcl scripts, 616, 618–620
  Puppet, 607–610
    architectures, 607–610
    catalogs, 607
    classes, 607

        concepts, 607–608
        configuration script, 608–610
        example, 608–610
        manifests, 607
        modules, 607
        resources, 607
    SaltStack, 612
        architectures, 612–614
        configuration script, 614–615
        examples, 614–615
        grains, 613
        masters, 612
        mines, 614
        minions, 612
        monitor (beacons), 613
        pillars, 614
        proxy minions, 613
        respond (reactor), 613
        SSH, 612
        states (formulas), 613
system priority, LACP, 103

T

tables
    BGP tables, 169
    IP routing tables, 169
    neighbor tables, 169
    TCAM tables
        Layer 2 switches, 17
        Layer 3 switches, 18
tabs (indentation), idiomatic Python (being Pythonic), 559
TACACS+, 497
    configuring, 501
    packet flow, 498
tail drops, 344–345
Task API, 589
tasks, Ansible, 604
TCA (Topology Change Acknowledgement) BPDU, 57
TCAM tables
    Layer 2 switches, 17
    Layer 3 switches, 18
tcl scripts, EEM, 616, 618–620
TCN (Topology Change Notification) BPDU, 57, 58
TCP (Transmission Control Protocol)
    congestion avoidance, 344–345
    global synchronization, 345
Template Editor, DNA Center, 547–548
Template Programmer API, 589
templates, SDM templates, 30–31
testing connectivity, ping command, 353
threats, network security, 466–468
threatscape, network security, 465–468
three-tier design, 5–6
tiered hub-and-spoke topologies, 233
time packet exchanges, NTP, 219
timers
    HSRP, 190–191
    OSPF, 136–137
TLOC (Transport Location), SD-WAN, 297, 299, 301
topological (link-state) databases, 122
topologies
    changes, RSTP, 67–68
    changes, BPDU, 58
        direct topology changes, 58–59
        indirect topology changes, 59
        insignificant topology changes, 59
    DMVPN, 234
    fully meshed network topologies, 233
    hub-and-spoke network topologies, 233
    joined hub-and-spoke topologies, 233
    MST, 89
    partial mesh topologies, 233
    redundant topologies
        broadcast storms, 56
        MAC database instability, 56
        multiple frame transmission, 56
    remote SPAN, 369
    Review Lab 1, 621
    Review Lab 2, 633
    SD-Access, 282–283
    site-to-site VPN, 232–233
    tiered hub-and-spoke topologies, 233
    VTP topology example, 47–48
TPID (Tag Protocol Identifiers), 40
traceroute command, 355–357
tracking objects, HSRP, 191–194
traffic patterns, VLAN, 33–34
traffic sniffers, 365
traffic starvation, 342

transparent mode (VTP), 47

transport mode, IPsec VPN, 235–236
  GRE over IPsec transport mode, 242
  IPsec over GRE tunnel mode, 242–243

troubleshooting
  basic steps, 350
  concepts, 349–350
  connectivity testing, 353, 356–357
  debug command, 357
    access lists, 359–360
    conditional debug command, 358–359
  diagnostics, 350–351
    debug command, 357–360
    ping command, 352–355
    traceroute command, 355–357
  IOS IP SLA, 360–365
  logging services, 373–374
    Flexible NetFlow, 384–387
    NetFlow, 378–387
    SNMP, 376–378
    syslog, 374–376
  ping command, 352
    connectivity testing, 353
    extended ping command, 353–355
    options, 353
  process, 351–352
  SD-WAN troubleshooting tools, 591
  SPAN, 365–366
    ERSPAN, 370–372
    local SPAN, 366–368
    overview, 365–366
    remote SPAN, 368–370
  traceroute command, 355–357
  traffic sniffers, 365

trunk mode, switchport mode trunk command, 44

trunking
  DTP, 45
    combinations, 45
    configuring, 45–46
    verifying status, 44–46
  verifying trunk status with DTP, 45–46
  VTP, 46–47
    advertisements, 47
    client mode, 47
    configuration example, 49
    configuration revision, 48
    off mode, 47
    server mode, 47
    show VTP status command, 49–50
    topology example, 47–48
    transparent mode, 47
    verifying, 49–50
    versions, 48–49

trust boundaries, QoS, 333–334

TrustSec, 276, 478–482

tuning NAT, 217–218

tunnel mode, IPsec VPN, 236

tunneling
  ETR, 254
  GRE over IPsec, Tunnel IPsec Profiles, 245–246
  guest tunneling (auto-anchor mobility), 441–442, 458
  ITR, 253–254
  PETR, 255
  PITR, 255
  split tunneling, 416
  VTI, 243, 247–249

two-tier (collapsed core) design, 4–5

Tx beamforming, 403–404

Type of Service, QoS, 337–338

U

UDLD (Unidirectional Link Detection), 82–83

UDP, multicasting, 306

underlay networks, 227, 272–273
  fabric technology, 274
  SD-WAN, 287

unequal-cost load balancing, EIGRP, 118–119

unicasting
  bandwidth, 306
  multicasting versus, 304–305
  traffic streams, 304–305

unknown unicast flooding, 15

UPDATE messages, BGP, 170

update packets, 108

updates, batched updates, BGP, 165

URL filtering, 478

user authentication, 802.1X, 449–451, 453–457

usernames, device access security, 494–495

V

vAnalytics, 293–294

vBond orchestrator, 288–289

vectors of data loss and exfiltration, 467

verifying
- 802.1Q trunk ports, 41–43
- BPDU Guard, 78–79
- conditional debug command, 358
- DTP, status, 44–45
- EBGP, 178–182
- ERSPAN, 372
- EtherChannel, 97, 99–101
- Flexible NetFlow, 386–387
- GRE, 230–232
- local SPAN, 367–368
- metrics (K values), 111–112
- MST, 88
- multiarea OSPF functionality, 138–141
- NetFlow, 382–384
- NTP
  - *associations, 225–226*
  - *status, 224–225*
- OSPF functionality, 138–141
- OSPFv3, 161–162
- remote SPAN, 369–370
- RSTP, bridge ID, 69–70
- STP mode, 76–77
- STP PortFast, 78–79
- trunk status with DTP, 45–46
- UDLD, 82–83
- VRF-Lite, 531–532
- VTP, 49–50
- wide metrics, EIGRP, 114–115

VID (VLAN Identifiers), 40

virtual interfaces, NAT, 213–214

virtual networks, mapping users to, 278

virtualization, 517
- data path virtualization, 527, 528
- device virtualization, 527
- DNA Center, 536
- Enterprise NFV, 523
  - *architectures, 523–524*
  - *NFVIS, 524–527*
- hypervisors, 520–522
- Layer 2 virtualization, 527–529
- Layer 3 virtualization, 527–529
- network path isolation, 527
- servers, 517–520
- VM, 518–519
  - *defined, 522*
  - *managing, 522–523*
- vNIC, 518–519
- VRF, 529–530
- VRF-Lite, 527–528
  - *configuring, 530–531*
  - *verifying, 531–532*

VLAN (Virtual Local Area Networks), 40–41. *See also* LAN
- 802.1Q trunk ports, 38–39
  - *configuring, 41*
  - *tagging process, 39–40*
  - *verifying, 41–43*
- 802.1X endpoint authentication, 487
- access ports, 35–36
- allowed VLAN, switchport trunk native vlan command, 41
- creating, 35
- DTP, 44–45
  - *combinations, 45*
  - *configuring, 45–46*
  - *verifying status, 44–46*
- end-to-end VLAN, 34
- inter-VLAN routing, 34, 50–51
  - *encapsulation dot1q command, 51–52*
  - *external routers, 51–52*
  - *subinterfaces, 51–52*
  - *SVI, 52–53*
- Layer 2 virtualization, 527–529
- local VLAN, 34–35
- native VLAN, 40, 41
- overview, 33–35
- routed switch ports, 53–54
- show interfaces switchport command, 38, 41–45
- show interfaces trunk command, 41–43
- show vlan brief command, 36–37
- show vlan command, 36–37
- show vlan id 2 command, 36–37
- show vlan name IT command, 36–37
- switchport mode dynamic auto command, 44
- switchport mode dynamic desirable command, 44
- switchport mode trunk command, 44
- switchport trunk encapsulation dot1q command, 41
- traffic patterns, 33–34

- VID, 40
- VTP, 46–47
  - *advertisements, 47*
  - *client mode, 47*
  - *configuration example, 49*
  - *configuration revision, 48*
  - *off mode, 47*
  - *server mode, 47*
  - *show VTP status command, 49–50*
  - *topology example, 47–48*
  - *transparent mode, 47*
  - *verifying, 49–50*
  - *versions, 48–49*

VM (Virtual Machines), 518–519
- defined, 522
- managing, 522–523

vManage, 288–289, 297
- GUI, 289
- high resiliency mode, 290
- multitenant mode, 290
- NMS, 288
- Postman, REST API operations, 596–598
- Python, REST API operations, 594–596
- RBAC, 289
- REST API, 590–598

vNIC (Virtual Network Interface Cards), 518–519

VPN (Virtual Private Networks), 469–470
- DMVPN, 234
- FlexVPN, 235
- GRE over IPsec transport mode, 242
- IPsec over GRE tunnel mode, 242–243
- IPsec VPN, 235–236, 238–239
- SD-WAN, 298–299
- site-to-site VPN, 232
  - *configuring, 242–249*
  - *DMVPN, 234*
  - *FlexVPN, 235*
  - *fully meshed topologies, 233*
  - *GRE over IPsec Site-to-Site VPN, 243–244*
  - *hub-and-spoke topologies, 233*
  - *individual point-to-point VPN connections, 232–233*
  - *joined hub-and-spoke topologies, 233*
  - *partial mesh topologies, 233*
  - *tiered hub-and-spoke topologies, 233*
  - *topologies, 232–233*
  - *VTI over IPsec, 247–249*

VRF (Virtual Routing and Forwarding), 529–530, 636–637

VRF-Lite, 527–528
- configuring, 530–531
- verifying, 531–532

VRRP (Virtual Router Redundancy Protocol), 185, 199–200
- authentication, 201
- configuring, 201–203

VSL (Virtual Switched Links), VSS, simplified campus design, 9

vSmart controllers, 288–291, 297–298, 301

VSS (Virtual Switching System), simplified campus design, 9–11

VTI (Virtual Tunnel Interfaces), 243, 247–249

VTP (VLAN Trunking Protocol), 46–47
- advertisements, 47
- client mode, 47
- configuring
  - *example, 49*
  - *configuration revision, 48*
- off mode, 47
- server mode, 47
- show VTP status command, 49–50
- topology example, 47–48
- transparent mode, 47
- verifying, 49–50
- versions, 48–49

VTY ports, syslog messages, 374

vulnerabilities, network security, 466

VXLAN (Virtual Extensible Local Area Networks), 251, 262–263
- encapsulation, 263–265
- encapsulation/decapsulation, 278
- gateways, 251
- VXLAN-GPO headers, 266–267

W

WAN (Wide Area Networks)
- SD-WAN, 285–294
  - *administration, 591*
  - *analytics, 293–294*
  - *API library/documentation, 592–598*
  - *application performance optimization, 294–295*

    - *application-aware routing, 294*
    - *automation, 293–294*
    - *certificate management, 591*
    - *color, 297–298*
    - *components, 287–294*
    - *configuration, 591*
    - *control plane, 288, 290–291*
    - *data plane, 291–292*
    - *device inventories, 591*
    - *IaaS, 295*
    - *management plane, 289–290*
    - *monitoring, 591*
    - *need for, 286–287*
    - *OMP, 298–301*
    - *orchestration plane, 288–289*
    - *organization names, 296–297*
    - *overlay networks, 287*
    - *private IP addresses, 297*
    - *public IP addresses, 297*
    - *QoS, 294*
    - *real-time monitoring, 591*
    - *REST API, 290, 590–598*
    - *SaaS, 295*
    - *service routes, 301*
    - *site ID, 296*
    - *solution example, 295–296*
    - *system IP addresses, 296*
    - *TLOC, 297, 299, 301*
    - *troubleshooting tools, 591*
    - *underlay networks, 287*
  - WAN Edge cloud routers, 287
  - WAN Edge routers, 287–292, 297–298, 301

watts (W), 395, 396

wavelengths, RF communications, 391

web portals, external authentication, 459

WebAuth (Web Authentication)
  - configuring, 459–463
  - device registration, 490
  - EasyConnect, 490
  - guest access, 457–458
    - *auto-anchor mobility (guest tunneling), 458*
    - *centralized web authentication, 459*
    - *configuring WebAuth, 459–463*
    - *local web authentication, 458*
    - *local web portal, external authentication, 459*
  - identity management, 489–490
  - NAD
    - *Central WebAuth, 489*
    - *Wired NAD with Local WebAuth, 489*

well-known BGP attributes, 175
  - discretionary attributes, 175
  - mandatory attributes, 175

WFQ (Weighted Fair Queuing), 342–344

whitespace, idiomatic Python (being Pythonic), 559

wide metrics, EIGRP, 112–115

Wi-Fi
  - antennas
    - *directional antennas, 430–431*
    - *EIRP, 398–399*
    - *isotropic antennas, 397–399*
    - *omnidirectional antennas, 429–430*
    - *RF communications, 397–399*
    - *types, 428–431*
  - AP
    - *802.11ac MIMO, 401–404*
    - *802.11ac MU-MIMO, 405–406*
    - *802.11n MIMO, 400–404*
    - *CAPWAP, 413*
    - *split MAC, centralized WLC deployments, 411–413*
    - *wireless-to-wired AP traffic flows, 409*
    - *wireless-to-wireless AP traffic flows, 410*
    - *WLC discovery process, 422–425*
  - client authentication
    - *802.1X user authentication, 449–451, 453–457*
    - *EAP, 452–457*
    - *guest access with WebAuth, 457–463*
    - *PSK authentication, 446–449*
    - *WPA, 445–449*
  - IEEE wireless standards, 399
    - *802.11a wireless standard, 400*
    - *802.11ac wireless standard, 400*
    - *802.11ax (Wi-Fi 6) wireless standard, 401*
    - *802.11b wireless standard, 399–400*
    - *802.11g wireless standard, 400*
    - *802.11n wireless standard, 400*
  - location services, 442
    - *accuracy, 444–445*
    - *CMX, 442–444*

- MIMO, 400
  - *802.11ac MIMO, 401–405*
  - *802.11n MIMO, 400–405*
  - *MU-MIMO, 400–401, 405–406*
- mobility groups, 435–436
- networks, mapping markings, 340
- OFDM, 400
- RF communications, 389
  - *amplitude, 391–392*
  - *antennas, 397–399*
  - *decibels (dB), 395–397*
  - *EIRP, 398–399*
  - *free path loss, 392–393*
  - *frequencies, 390*
  - *noise, 394*
  - *RCPI, 394*
  - *RSSI, 393–394, 434*
  - *SNR, 394–395*
  - *spectrums, 389–390*
  - *watts (W), 395, 396*
  - *wavelengths, 391*
- roaming, 434
  - *active scanning, 435*
  - *auto-anchor mobility (guest tunneling), 441–442*
  - *background scanning, 435*
  - *broadcast probes, 435*
  - *directed probes, 435*
  - *Layer 2 roaming, 437–439*
  - *Layer 3 roaming, 439–441*
  - *maximum data retry counts, 434*
  - *mobility groups, 435–436*
  - *passive scanning, 435*
  - *proprietrary load-balancing, 434–435*
  - *on-roam scanning, 435*
  - *RSSI, 434*
  - *SNR, 434*
  - *types, 436–441*
- Wi-Fi 6, 401
- WLAN
  - *autonomous AP deployments, 408–410*
  - *centralized WLC deployments, 408, 410–415*
  - *cloud-managed deployments, Meraki, 408, 416–417*
  - *FlexConnect deployments, 408, 415–416*
  - *SD-Access deployments, 408*
- WLC
  - *AP discovery operation, 422–425*
  - *AP modes, 425–428*
  - *Catalyst 9800 Series wireless controllers, 417–420*
  - *central switching, 416*
  - *centralized deployments, 408, 410–415*
  - *Cisco Mobility Express, 422*
  - *fabric technology, 277, 279*
  - *local switching, 416*
  - *split tunneling, 416*
  - *WebAuth, 489*

wildcard masking, ACL, 505–507

Wired NAD with Local WebAuth, 489

wireless technology. ***See*** **Wi-Fi**

wireless-to-wired AP traffic flows, 409

wireless-to-wireless AP traffic flows, 410

WLAN (Wireless LAN)
- autonomous AP deployments, 408–410
- centralized WLC deployments, 408
  - *CAPWAP, 413*
  - *split MAC, 411–413*
  - *traffic flows, 413–415*
- cloud-managed deployments, Meraki, 408, 416–417
- FlexConnect deployments, 408, 415–416
- SD-Access deployments, 408

WLC (Wireless LAN Controllers)
- AP discovery operation, 422
  - *AP failovers, 423–425*
  - *AP join orders, 423*
  - *AP modes, 425–428*
  - *DHCP, 423*
  - *DNS, 423*
  - *IPv4 addressing, 422*
  - *IPv6 addressing, 422*
  - *subnetwork broadcasts, 422*
- Catalyst 9800 Series wireless controllers, 417–420
  - *cloud-managed deployments, 419*
  - *embedded controllers, 419–420*
- central switching, 416
- centralized deployments, 408
  - *CAPWAP, 413*
  - *split MAC, 411–413*
  - *traffic flows, 413–415*
- Cisco Mobility Express, 422
- local switching, 416
- split tunneling, 416
- WebAuth, 489

WPA (Wireless Protected Access), 445–449

writing
- policies, EEM, 617–620
- Python scripts, 557

X

XML (Extensible Markup Language), 570
  namespaces, 571–572
  NETCONF, XML encoding, 579
  Python, XML example, 570–571

Y - Z

YAML files, Ansible, 602
YANG (Yet Another Next Generation), 573–574 cip